Ermittlung von Performancewerten für die WinCC Prozesskommunikation zu verschiedenen Steuerungen

SIMATIC S7-300/400 & WinCC V7.0

Bachelorarbeit • 2012/2013

Applikationen & Tools

Answers for industry.

SIEMENS

HOCHSCHULE MITTWEIDA
UNIVERSITY OF APPLIED SCIENCES

SIEMENS

BACHELORARBEIT

Herr
Min Xie

Ermittlung von Performancewerten für die WinCC Prozesskommunikation zu verschiedenen Steuerungen

2012 / 2013

Fakultät: Maschinenbau

BACHELORARBEIT

Ermittlung von Performancewerten für die WinCC Prozesskommunikation zu verschiedenen Steuerungen

Autor:
Herr Min Xie

Studiengang:
Mechatronik, B.Eng.

Seminargruppe:
ME09w1B

Erstprüfer:
Prof. Dr.-Ing. Swen Schmeißer

Zweitprüfer:
Dipl.-Ing. Farzad Zare

Einreichung:
Nürnberg, 27.02.2013

Verteidigung/Bewertung:
Mittweida, 2013

Faculty of Mechanical Engineering

BACHELOR THESIS

acquisition and evaluation of performance data for process communication in WinCC with different S7-controllers

author:
Mr. Min Xie

course of studies:
Mechatronics, B.Eng.

seminar group:
ME09w1B

first examiner:
Prof. Dr.-Ing. Swen Schmeißer

second examiner:
Dipl.-Ing. Farzad Zare

submission:
Nürnberg, 27.02.2013

defence/evaluation:
Mittweida, 2013

Danksagung

Diese Bachelorarbeit ist während des Wintersemesters 2012/2013 in der Siemens AG, in dem Industry Sector, Industry Automation Division, I IA AS S SUP FA Nürnberg entstanden.
Dabei hatte ich die Gelegenheit, das beim Studium gelernte Wissen in die Praxis umzusetzen, und mehr Kenntnisse über die Automatisierungstechnik zu bekommen.
Ich möchte mich bei allen Ansprechpartnern für die fachliche Unterstützung herzlich bedanken. Sie sind Herr Farzad Zare, der Consultant von WinCC Classic; Herr Jürgen Bohrer, der Gruppenleiter von SUP FA 2 (Visualizations); und die andere Kolleginnen und Kollegen von der Abteilung I IA AS S SUP FA.
Mein besonderer Dank gilt Herr Dr.-Ing Jun Yao und Herr Zhongyang Wang, die mir die Chance gegeben haben, eine tiefere Qualifikation bei Siemens durch eine Bachelorarbeit durchzuführen.
Weiterhin möchte ich mich bei Herrn Prof. Dr.-Ing. Swen Schmeißer für die Betreuung der Bachelorarbeit und die freundliche Unterstützung bedanken.

Inhaltverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Diagrammverzeichnis

1 Übersicht

Im eröffnenden Kapitel werden die Motivation und die Aufgabenstellung dieser Bachelorarbeit erläutert. Gleichzeitig erfolgt eine kurze Übersicht zu den einzelnen Kapiteln dieser Arbeit.

1.1 Motivation

Ein Automatisierungsprojekt erfordert häufig einen intensiven Datenaustausch zwischen einer Steuerung (SIMATIC S7-Steuerungen) und einem SCADA (Supervisory Control and Data Acquisition) System. Als SCADA System zur Visualisierung, Bedienung, Erfassung und Archivierung von Alarmen und Prozesswerte wird weltweit unter Anderem WinCC (Windows Control Center) eingesetzt.
Als zentrale Kommunikationskomponente zur Steuerungen, von allem SIMATIC Steuerungen hat sich WinCC durchgesetzt. In WinCC wird die Verbindung über verschiedene Kommunikationstreiber realisiert. Es steht eine Vielzahl an Kommunikationstreibern für die Anbindung verschiedener Automatisierungssysteme über verschiedene Bussysteme zur Verfügung.
Als Bedien und Beobachtung System der HMI (Human Machine Interface)-System spielt WinCC eine sehr wichtige Rolle. Die sichere und effiziente Kommunikationsperformance beeinflusst direkt auf die Arbeitseffizienz des Operators, sowie die Sicherheit des Herstellungsprozesses.
Bei der praktischen Anwendung in der industriellen Branche sind folgende Verhalten festgestellt wurden.

- Kommunikation ist unstabiel.
- Aktualisierung von Prozessvariablen sind langsam, oder werden nicht aktualisiert.
- Der Bildwechsel der WinCC Runtime Oberfläche ist langsam.

In dieser Bachelorarbeit wurde ein Simulationsprojekt in WinCC und der verschiedenen PLCs (Programmable Logic Controller) erstellt, um die realen Bedingungen in der industriellen Automatisierung zu simulieren. Danach wurde ein komplettes Messen durch die verschiedene Dimensionen und Gesichtspunkte zur der Kommunikationsperformance durchgeführt. Zuletzt wurden die Ergebnisse des Messens ausführlich zusammengefasst, um die optimale Methode zur Konfiguration und Projektierung zu erhalten bzw. zu definieren.

1.2 Aufgabenstellung

Im Rahmen der Bachelorarbeit sollen Performancewerte für WinCC mit S7-400 und S7-300 CPUs über verschiedenen Kommunikationskanälen ermittelt werden. Anhand der Messungen, unter der Berücksichtigung der Randbedingungen und Systemparameter, sollen dann Kriterien sowie Argumente für die folgenden Fragestellungen ausgearbeitet werden:

- Welche Kommunikationsart ist für das geplante Automatisierungsprojekt am besten geeignet?
- Mit welchen Daten-Übertragungszeiten ist bei typischen Konfigurationen und Systemen zu rechnen?
- Welche Randbedingungen sind zu berücksichtigen?

1.3 Kapitelübersicht

Diese Bachelorarbeit besteht aus 6 Kapiteln.

Kapitel 1 skizziert kurz Motivation und Aufgabenstellung sowie die Kapitelübersicht.

Im Kapitel 2 werden die Konzepte der Kommunikation zwischen WinCC und SPS (Speicherprogrammierbare Steuerung) dargestellt. Dabei werden die Einflussfaktoren der WinCC Kommunikationsperformance und Ablauf des Messprojektes vorgestellt.

Kapitel 3 erläutert die Topologie und Komponenten der Versuchsanlage.

Im Kapitel 4 werden die Projektierung der Simulationsumgebung

Danach beschreibt Kapitel 5, wie die Performancewerte und Einflussfaktoren gemessen und ermittelt werden?

Kapitel 6 fasst abschließend die Resultate der Bachelorarbeit zusammen. Zusätzlich werden die getroffene Probleme und mögliche Weiterentwicklungen gegeben.

2 Konzept

2.1 WinCC und Steuerungen im Automatisierungssystem

Die Automatisierungstechnik ist eine Einzelwissenschaft des Ingenieurwesens, die zum Ziel hat, Maschinen oder Anlagen zu automatisieren, also selbständig und ohne Mitwirkung von Menschen zu betreiben. Es ist eine Hilfsdisziplin für alle Bereiche der Technik. Die zugehörigen Methoden und Lösungen (inkl. Messen, Steuern, Regeln, Kommunikation, Mensch-Maschine-Schnittstelle, Sicherheit usw.) sind das Ergebnis einer verallgemeinerten (abstrahierenden) Modellbetrachtung realer physikalischer Systeme.

2.1.1 Automatisierungspyramide

Bei der praktischen Anwendung wird die Automatisierungstechnik als ein komplettes System aufgebaut. Das System kann mit der Automatisierungspyramide genau dargestellt werden.
Die Automatisierungspyramide dient der Einordnung von Techniken und Systemen in der Leittechnik und stellt die verschiedenen Ebenen in der industriellen Herstellung dar.

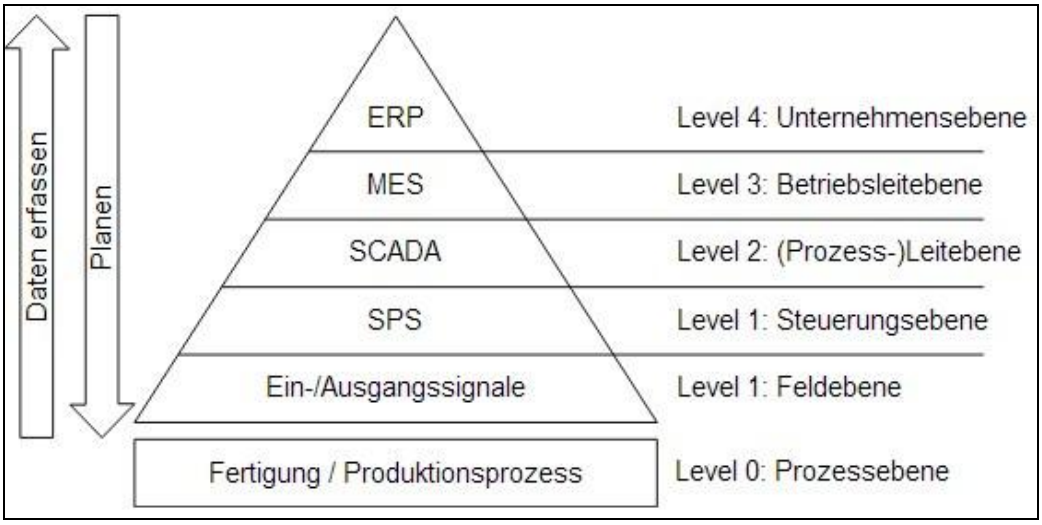


Abbildung 2-1: Automatisierungspyramide[1]

Jeder Ebene kommt eine eigene Aufgabe in der Produktion zu, wobei es je nach betrieblicher Situation fließende Grenzen gibt. Entsprechend der Aufgabe der Ebene haben sich spezifische Techniken der analogen als auch digitalen Datenübertragung und -verarbeitung entwickelt.
Level1: Steuerungsebene. Typische Produkte: S7-300/400 von Siemens
Level2: (Prozess-)Leitebene. Typische Produkte: WinCC von Siemens

[1] Quelle: Wikipedia Stichwort: Automatisierungspyramide

S7 Steuerung als ein System in der Feldebene beantwortet sie für die Bearbeitung und Ausführung der Prozesssignale von der Feldebene.
Als ein typisches SCADA System übernimmt WinCC die Aufgaben für die Überwachung, Beachtung, Zusammenfassung und Visualisierung der Prozessdaten. Es liest, überwacht, archiviert die Prozessdaten von Steuerungen und zeigt die als eine detaillierte und visualisierte Form im Bild an.
Durch Kombination und Integration zwischen SCADA und Steuerungen erstellt es ein einfaches und visualisiertes Interface zwischen Mensch und Maschine, sogenannt HMI (Human Machine Interface). Bei der modernen Industrieherstellung wird es in vielen Bereichen (z.B. Maschinenbau, Autoindustrie, Chemieindustrie usw.) angewendet.

2.1.2 Kommunikation zwischen WinCC und Steuerungen

Um die Kommunikationsperformance zu untersuchen, sollen wir zuerst die Grundsätze und die Struktur der WinCC Prozesskommunikation kennenlernen[2].

Datenmanager
Die Verwaltung des Datenbestandes übernimmt in WinCC der Datenmanager. Dieser ist für den Anwender jedoch nicht sichtbar. Der Datenmanager arbeitet mit den im WinCC Projekt erstellten und in der Projektdatenbank abgelegten Daten. Er übernimmt die komplette Verwaltung der WinCC Variablen während der Laufzeit von WinCC zur Runtime. Alle Applikationen von WinCC müssen die Daten beim Datenmanager in Form von WinCC Variablen anfordern. Diese Applikationen sind unter anderem die Graphics Runtime, die Alarm Logging Runtime und die Tag Logging Runtime.

Kommunikationsstruktur
Die Verwaltung der WinCC Variablen zur Laufzeit der Runtime wird vom WinCC Datenmanager durchgeführt. Die verschiedenen WinCC Applikationen richten ihre Variablenanforderungen an den Datenmanager.
Der Datenmanager hat nun die Aufgabe, die geforderten Variablenwerte vom Prozess zu holen. Dies erledigt er über den jeweiligen ins WinCC Projekt eingebundenen Kommunikationstreiber. Der Kommunikationstreiber bildet über eine seiner Kanal Units (Treiber) die Schnittstelle zwischen WinCC und dem Prozess. Der hardwaremäßige Anschluss zum Prozess wird im Großteil der Fälle über einen Kommunikationsprozessor realisiert. Über diesen schickt der WinCC Kommunikationstreiber nun Anforderungstelegramme an das Automatisierungsgerät. Dieses sendet die geforderten Prozesswerte in entsprechenden Antworttelegrammen an WinCC zurück. Die nachfolgende Abbildung zeigt die Kommunikationsstruktur zwischen WinCC und Steuerung.

[2] Die Inhalte dieses Abschnittes wurden von Siemens Handbuch WinCC Communication Manua Band 1 2.3.1 Seite 2-29 zitiert.

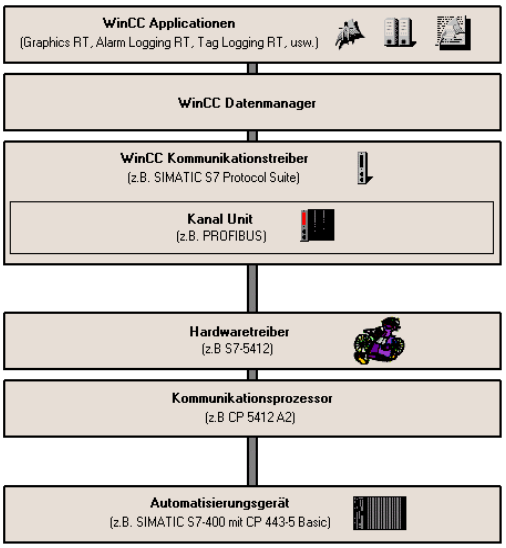


Abbildung 2-2: Kommunikationsstruktur zwischen WinCC und SPS[3]

<u>Kommunikationstelegramme</u>

Die Kommunikation zwischen WinCC und der Steuerung erfolgt über das Versenden von Telegrammen.

Anstatt des Begriffs Telegramm wird des Öfteren auch der Begriff PDU (Protocol Data Unit) verwendet.

Eine WinCC Station fordert zum Beispiel über das Senden eines Anforderungstelegrammes von einem Automatisierungsgerät bestimmte Daten an. Dieses sendet daraufhin die angeforderten Daten in Form eines Rücktelegrammes an die WinCC Station. Die nachfolgende Abbildung zeigt beispielhaft den Aufbau eines Telegramms.

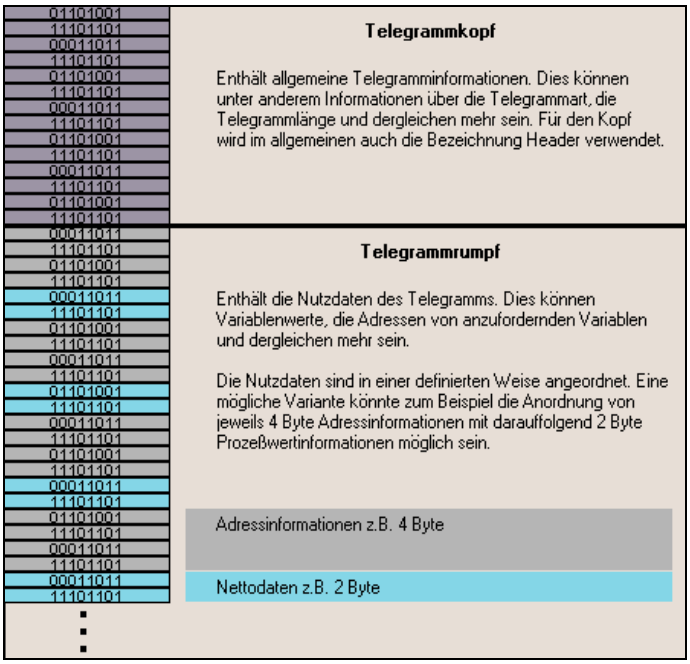


Abbildung 2-3: Aufbau des Kommunikationstelegramms[4]

[3] Quelle: Siemens Handbuch WinCC Communication Manua Band 1

2.2 Einflussfaktoren der Kommunikationsperformance von WinCC

In diesem Abschnitt werden die Einflussfaktoren der Kommunikationsperformance zwischen WinCC und Steuerungen dargestellt. Die sind auch die Parameter und Faktoren, welche im Test berücksichtigt und eingerichtet werden.

Ab.2-4 zeigt die Haupteinflussfaktoren, welche in dieser Bacholerarbeit untersucht werden. Sie werden in 4 Gruppen unterteilt.

1. Kommunikationsart (Protokoll): Die Messanlage wird mit verschiedenen Kommunikationsprotokollen aufgebaut, um den Einfluss vom Kommunikationsart auf die Kommunikationsperformance herauszufinden.

2. Lesedienstart: Lesedienstart ist eine Konfiguration von WinCC und SPS. Es liefert 3 Möglichkeiten der Variablenanforderung von WinCC. Sie wird in WinCC konfiguriert, aber die Funktionalität wird von der Steurungstyp abhängig bestimmt.

3. Einstellung in der SPS: Mit den Einstellungen in der CPU-Konfiguration und Simulationsprogramm (Simulationslast) wird untersucht, welche Parameter auf der SPS-Seite Einfluss auf die Kommunikation zwischen WinCC und Steuerung haben.

4. WinCC-Projektierung: Hier wird untersucht, welchen Einfluss die Art der Dynamisierungen in den WinCC Bildern, Anzahl der Objekte, Triggerarten, Tag Logging, Alarm Logging… auf die Kommunikationsperformance haben.

Die oben aufgezählten Punkte werden ausführlich im Kapitel 5 genau dargestellt.

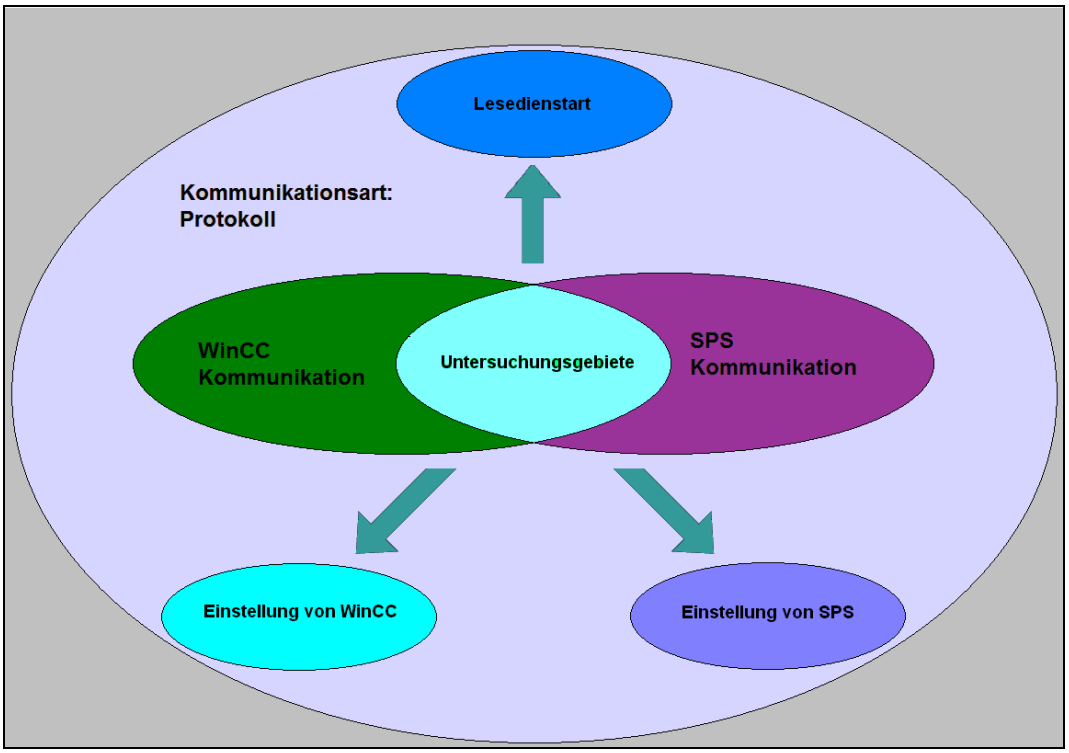


Abbildung 2-4: Einflussfaktoren zur Kommunikationsperformance von WinCC

[4] Quelle: Siemens Handbuch WinCC Communication Manua Band 1

2.3 Ablauf des Messprojektes

Nachdem die Einflussfaktoren festgesetzt worden sind, soll jetzt ein Plan zum Messen der Einflussparameter erstellt werden. Der Plan umfasst folgende Punkte:

- Auswahl von passender Hard- und Software
- Erstellung von einem Messprojekt (WinCC-Projekt und SPS Programm)
- Festlegung geeigneter Messmethoden
- Auswertung der Messergebnisse

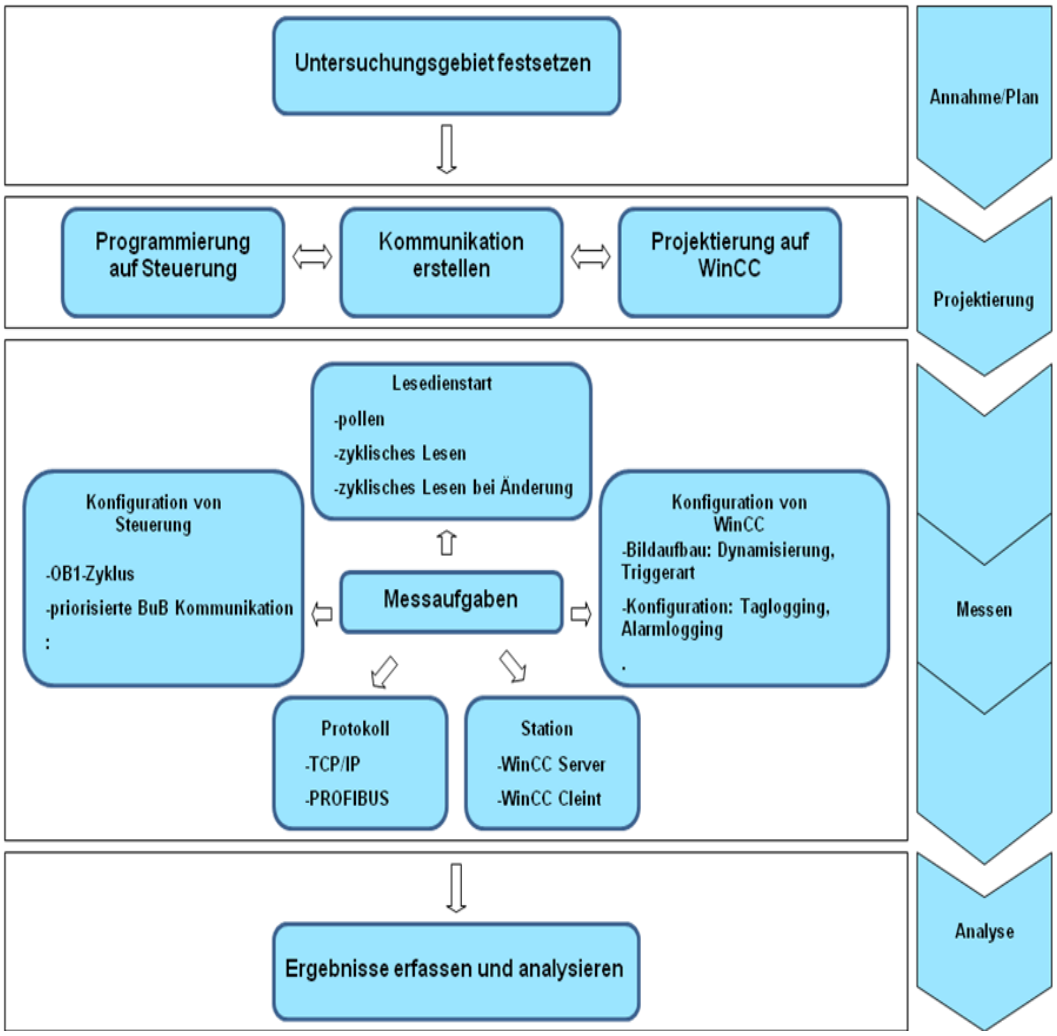


Abbildung 2-5: Fluss der Arbeit

3 Topologie und Komponentenlisten

In diesem Kapitel werden die Topologie der Testanlagen schematisch dargestellt und die verwendeten Komponenten in den Tabellen ausführlich aufgelistet.

3.1 Aufbau der Testanlage

Der Aufbau der Testanlage enthält folgende Komponenten:

Ein PG (Programmiergerät) als ES (Engineering Station)
Das PG wird für die Programmierung und Konfiguration des gesamten Projektes verwendet. Auf dem PG wird das SPS Programm und die WinCC Projekt vollständig erstellt und eingerichtet.
Das PG wird später für weiteren Test zusätzlich als WinCC Client verwendet.

Ein IPC (Industrie-PC) als WinCC Station
IPC ist robuster und hat eine höhere Leistung und Sicherheit als der normale PC, Er wird meistens in der Industriellen Umgebungen eingesetzt.
In dieser Arbeit wird das ganze Projekt (SPS und WinCC) auf der Engineering Station (ES) erstellt. WinCC Projekt ist in SIMATIC Manager integriert. Das Projekt wird von der ES auf die WinCC OS (WinCC Server) geladen.

Steuerungen S7-300/400
Um die Kommunikationsperformance zwischen WinCC und Steuerungen zu ermitteln und zu messen, sind Steuerungen natürlich unverzichtbar. Für den Test wurden 3 S7-300 und 2 S7-400 Steuerungen unterschiedlicher Typen ausgesucht. Die genauen Daten und Typen der CPUs und weitere Komponenten werden in den nachfolgenden Komponentenlisten aufgelistet.

Komponentenlist

Tabelle 3-1: S7-300 Baugruppen

Hardware	Anzahl	BestellNr.	Kommentar
CPU 315-2 PN/DP	1	6ES7 315-2EH14-0AB0	MMC siehe Zubehörlist
CPU 317-2 PN/DP	1	6ES7 317-2EK14-0AB0	MMC siehe Zubehörlist
CPU 319-3 PN/DP	1	6ES7 318-2EL01-0AB0	MMC siehe Zubehörlist
CP 343-1	1	6GK7 343-1EX30-0XE0	Kommunikationsmodul
Laststromversorgung PS 307	2	6ES7 307-1EA01-0AA0	für CPU 300 Baugruppen
PROFILSCHIENE L=480mm	1	6ES7 390-1AE80-0AA0	Rack für CPU 300 Baugruppen

Tabelle 3-2: S7-400 Baugruppen

Hardware	Anzahl	BestellNr.	Kommentar
CPU 412-2 PN	1	6ES7 412-2EK06-0AB0	MC siehe Zubehörlist
CPU 416-3 PN/DP	1	6ES7 416-3ES06-0AB0	MC siehe Zubehörlist
CP 443-1	1	6GK7 443-1EX20-0XE0	MC siehe Zubehörlist
Stromversorgungsbaugruppen PS 407	2	6ES7 407-0DA02-0AA0	für CPU 400 Baugruppen
UR2,Zentral-/Erweiterungsgerät	1	6ES7400-1JA01-0AA0	Zentralrack 1 für CPU 400; 9 Steckpl., K-Bus

Tabelle 3-3: IPC Baugruppen (als WinCC Server)

Hardware	Anzahl	BestellNr.	Kommentar
IPC 547D	1	6AG4 104-2DB41-0XX0	
CP 1628	1	6GK1 162-8AA00	
CP 5623	1	6GK1 562-3AA00	
Software/Lizenz		BestellNr.	
HARDNET-IE S7 V8.1	1	6GK1 716-1CB08-1AA0	
HARDNET-PB S7 V8.1	1	6GK1 713-5CB08-1AA0	
WinCC/Server (WinCC V7.0)	1	6AV6371-1CA07-0AX0	
1500 Achivvariablen (WinCC V7.0)	2	6AV6371-1DQ17-0AX0	Lizenz
8192 PowerTags (RT 8192)	1	6AV6381-2BH07-0AX0	

Tabelle 3-4: PG M3 Baugruppen (als Engineering Station + WinCC Client)

Hardware	Anzahl	BestellNr.	Kommentar
Field PG M3	1	6ES7715-1BB20-0AE2	
Software/Lizenz		BestellNr.	
8192 PowerTags (RC 8192)	1	6AV6381-2BS07-0AX0	
STEP7 V5.5 SP1	1	6ES7810-4CC10-0YA5	

Tabelle 3-5: Zubehöre

Hardware	Anzahl	BestellNr.	Kommentar
SIMATIC.NET, SCALANCE X208	1	6GK5208-0BA10-2AA3	
Micro Memory Card	5	6ES7953-8LP11-0AA0	8MByte für CPU300
RAM Memory Card	1	6ES7952-1AP00-0AA0	lang; 8MByte für CPU400

3.2 Netzverbindung der Testanlage

In diesem Abschnitt wird die Netzverbindung der Testanlage dargestellt. Die ES (Field PG) und WinCC OS (IPC) sind über eine Ethernet Netzwerkkarte verbunden. Diese dient für das Laden des Projektes von der ES auf die OS und später als Verbindung zwischen WinCC Client (Field-PG) und WinCC Server (IPC). Die Verbindung zwischen WinCC Server und Steuerungen wird nach den Arten der Übertragungsmedien eingeordnet.

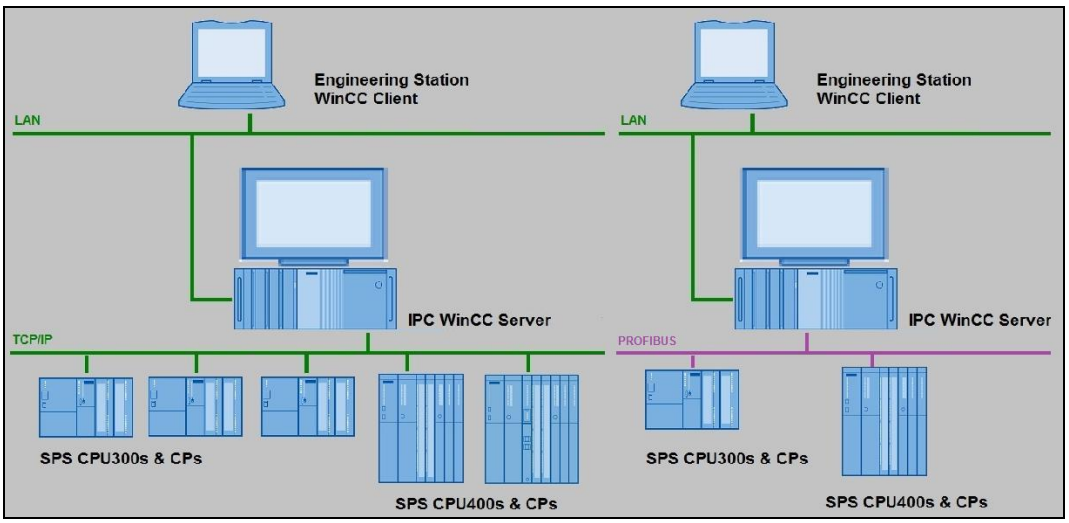


Abbildung 3-1: Schematische Netzwerkübersicht

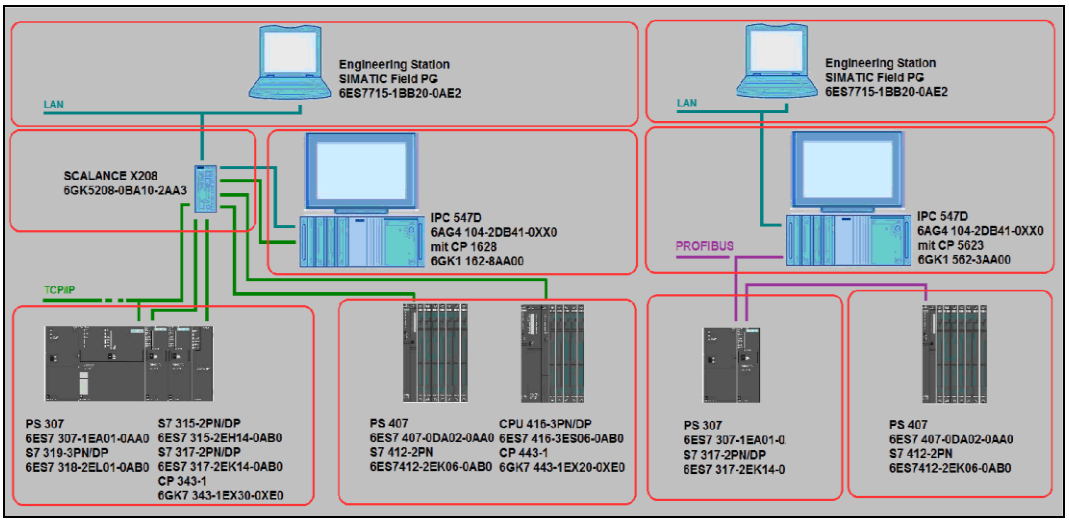


Abbildung 3-2: Topologie der Testanlage (TCP/IP und PROFIBUS)

4 Projektierung der Simulationsumgebung

4.1 Programmierung mit STEP7 zur Steuerung

In diesem Abschnitt wird die Hardwarekonfiguration und die Programmierung zur Steuerung S7-317 2PN/DP Schritt für Schritt erläutert.
Die Konfiguration und Programmierung der anderen CPUs ist ähnlich.

4.1.1 Hardwarekonfiguration

1. Ein neues STEP7 Projekt anlegen (Datei -> Neu -> Projektname: „WinCC Perfor mance“ eingeben).

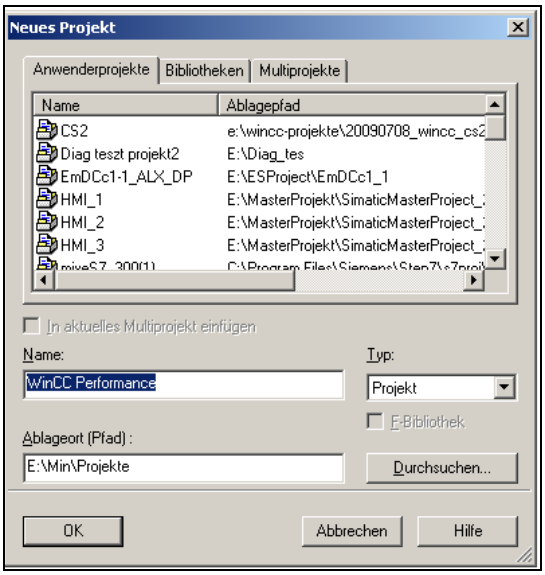


Abbildung 4-1: Ansicht des Fensters von Projekterstellung

2. Hardware konfigurieren („Hardware“ -> die verwendeten Komponenten nach Bestellungsnummer und Version in der rechten Reihe suchen und in richtigen Positionen einfügen)

Rack: Rack 300 Profilschiene
PS: PS 307 5A (PS - Powerstation)
CPU: CPU 317-2 PN/DP
(Bestellungsnummer: 6ES7 317-2EK14-0AB0, Version: 3.2)

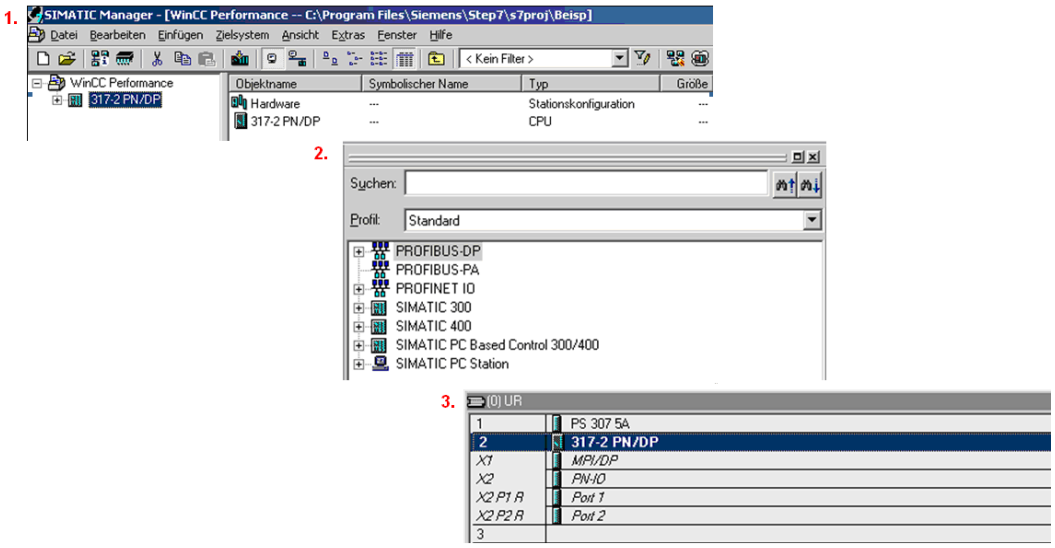


Abbildung 4-2: Hardwarekonfiguration

3. Allgemeine Kommunikationseinrichtung („PN-IO“ im Rack klicken -> Gerätename „S7-317“ eingeben -> „Eigenschaften“ -> „Erthernet(1)“ -> IP-Adresse und Subnetzmaske eingeben -> „OK“)

IP - Adresse: 172.16.70.10

Subnetzmaske: 255.255.0.0

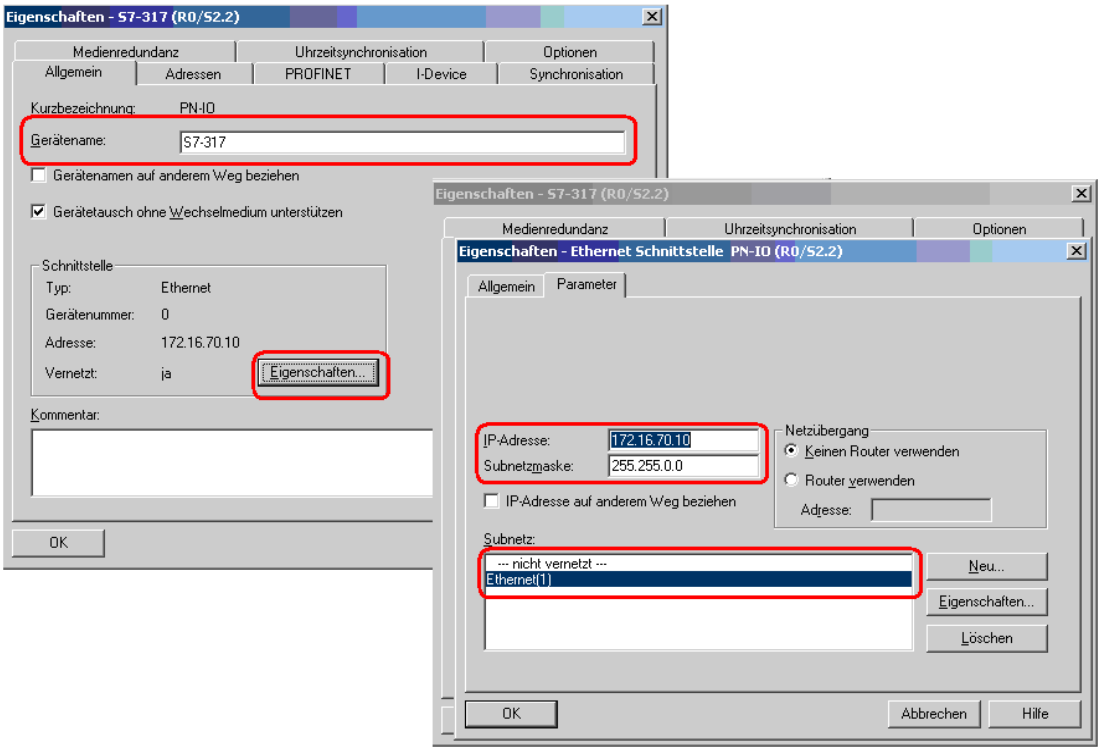


Abbildung 4-3:Kommunikationseinrichtung von CPU317-2PN/DP

4.Button  in der Menüzelle klicken, die Konfiguration speichern und laden

4.1.2 Konfiguration von 5 SPS Stationen mit STEP7

Durch die gleiche Methode werden die anderen 4 SPS Stationen erstellt, dabei wird auch die Verbindung mit TCP/IP konfiguriert. Die alle Datenbausteine können einfach von S7-317 2PN/DP kopiert werden.

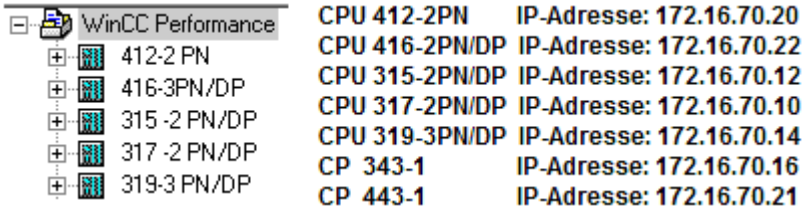


Abbildung 4-4: Integration der 5 SPS-Stationen

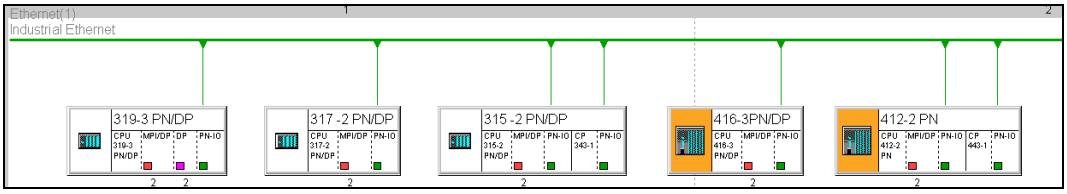


Abbildung 4-5: Netzverbindung der 5 SPS-Stationen

4.1.3 Programmierung

Die SPS-Programme sollen 2 Hauptfunktionen haben:

1. Simulation der Industrieumgebung
2. Durch die Einstellung verschiedener Parameter im Programm können die verschiedenen Messumgebungen eingestellt werden, sodass die Einflussfaktoren und die Randbedingungen herausgefunden werden können.

Die nachfolgenden Tabellen zeigen die Funktionen sowie wichtige Informationen dieser Bausteine an. Die SPS-Programme siehe STEP7 Projekt.

Tabelle 4-1: Funktionen von OB, FC, FB

	Funktion	Kommentar
OB1	FC3, FC 5 und FC6 aufrufen	
OB32	FC2, FC5 aufrufen	Alarmwerk kann man manuell einstellen.
FC1	Funktion für Alarm Logging	Zyklisches Bit Melden
FC2	Funktion für Tag Logging und Prozessvaribalen von Trigger	FB100, FB300 aufrufen
FC3	Funktion für Alarm Logging	FC1 aufrufen
FC5	Wert der Variablen (Sting) eingeben	Wert Var.0 vom DB31 eingeben
FC6	OB1 Zyklus einstellen	FB69 aufrufen
FB69	Simulation der Zyklusbelastung des OB1	OB1 belasten
FB100	Zyklisches Hochzählen von 0 bis 1000	Variablen inkrementieren sich.
FB300	Zyklisches Hochzählen von 0 bis 1000	Variablen inkrementieren sich.

Tabelle 4-2: Datei von DB, SFC

	Datentyp	Variable Anzahl pro DB	Funktion
DB1-DB10	ARRAY[1..4] DWORD	4	Prozessvariablen von WinCC Alarm Logging (Meldungen)
DB11-DB20	ARRAY[1..50] DWORD	50	Prozessorvariablen von WinCC Tag Logging (Archivierung)
DB21-DB30	ARRAY[1..100] DWORD	100	Prozessvariablen von WinCC Triggere
DB31-DB34[5]	STRING[200]	250	Prozessvariablen für Test der Kommunikationslast
DB222	STRING[200]	freiwillig eingeben	Text Bibliothek
DB69	-	-	Instanzdatenbaustein zu FB69
DB100	-	-	Instanzdatenbaustein zu FB1
DB300	-	-	Instanzdatenbaustein zu FB3
SFC20	-	-	Siehe Hilfe zu STEP7
SFC64	-	-	Siehe Hilfe zu STEP7

FC6, FB69, DB69, SFC20

Vor allem wird der Ablauf der zyklischen Programmbearbeitung vorgestellt. Tab.4-3 und Ab.4-6 zeigen die Phasen der zyklischen Programmbearbeitung. In der Ab.4-6 kann man sehen, dass jeder Zeitscheibe aus Betriebssystem, Anwenderprogramm und Kommunikation aufgebaut werden. Mit dem Funktionblock FB69 wird der OB1 Zyklus verlängert. In FC6 wird FB69 aufgerufen. Durch FC6 kann der OB1 Zyklus manuell eingestellt werden.

Tabelle 4-3: Zyklische Programmbearbeitung[6]

Schritt	Ablauf
1	Das Betriebssystem startet die Zykluszeitüberwachung.
2	Die CPU schreibt die Werte aus dem Prozessabbild der Ausgänge in die Ausgabebaugruppen.
3	Die CPU liest den Zustand der Eingänge an den Eingabebaugruppen und aktualisiert das Prozessabbild der Eingänge.
4	Die CPU bearbeitet das Anwenderprogramm in Zeitscheiben und führt die im Programm angegebenen Operationen aus.
5	Am Ende eines Zyklus führt das Betriebssystem anstehende Aufgaben aus, z. B. Laden und Löschen von Bausteinen.
6	Anschließend kehrt die CPU zum Zyklusanfang zurück und startet erneut die Anschließend kehrt die CPU zum Zyklusanfang zurück und startet erneut die Zykluszeitüberwachung.

[5] Bei S7-300 Serien darf nur von DB31 bis DB32 eingefügt werden, da die Plätze der Speicherkarte nicht ausreichend sind.

[6] Quelle: Siemens Handbuch S7-300 CPU 31xC und CPU 31x: Technische Daten Seite 177

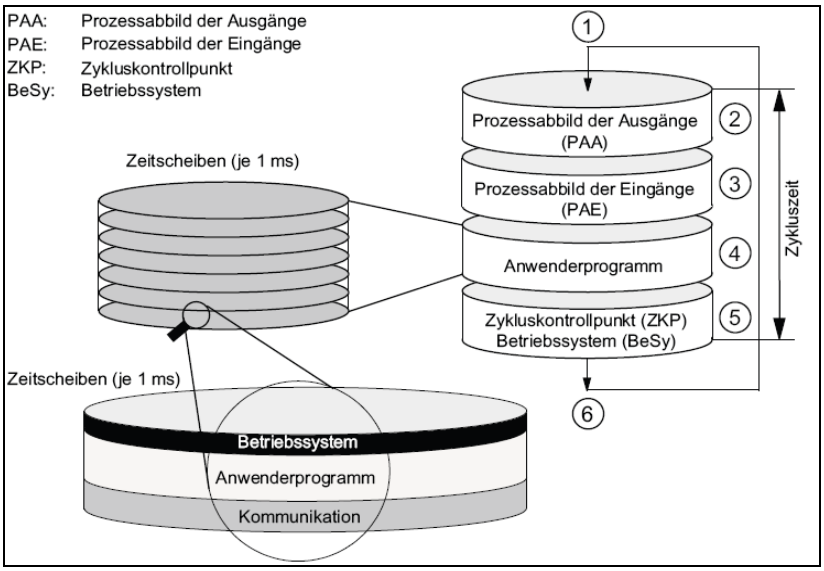


Abbildung 4-6: Ablauf der zyklischen Programmbearbeitung[7]

Klicken das Button „Online <-> offline“ , kann der Online Status von CPU im Dialogfeld gesehen werden. Im Dialogfeld wird die Zykluszeit von OB1 angezeigt. Die max. Überwachungszeit (horizontale Axe) ist 150 ms. Die grüne Axe zeigt der aktuelle OB1 Zyklus. Wenn man den Button „Aktualisieren“ klickt, wird die Zykluszeit aktualisiert.

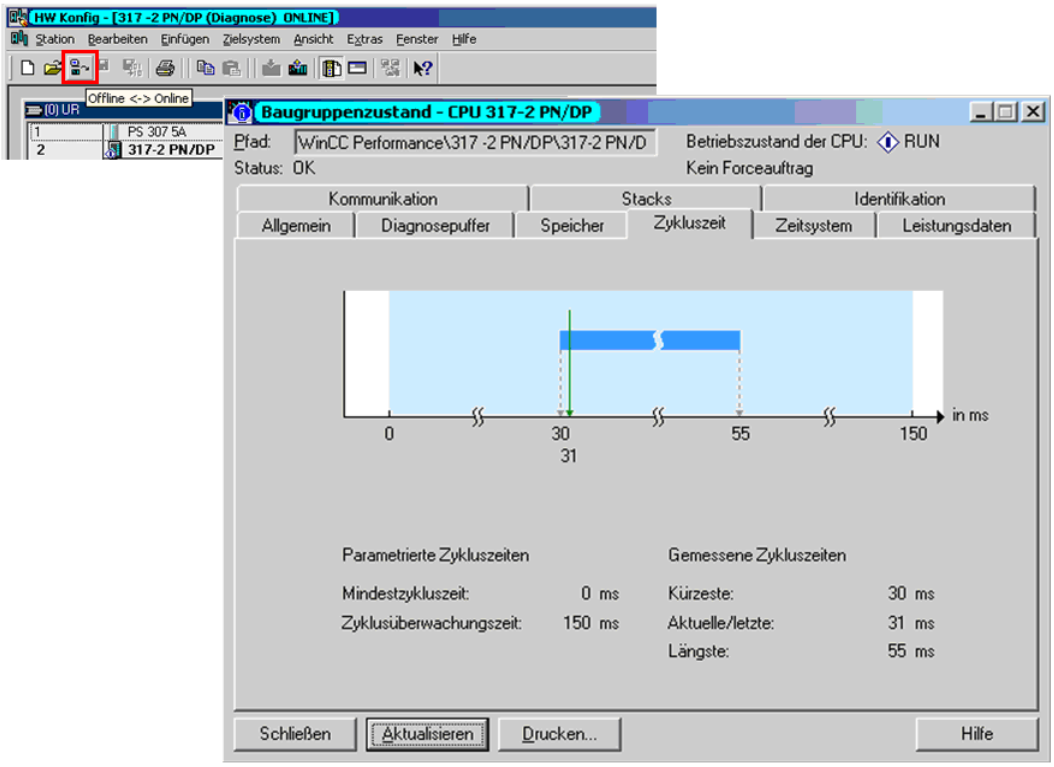


Abbildung 4-7: Überwachung des OB1 Zykluses

[7] Quelle: Siemens Handbuch S7-300 CPU 31xC und CPU 31x: Technische Daten

FC1, FC3
Simulation für Alarm Logging
Bit Meldung

FB100, DB100, FC2
Simulation für Tag Logging
Inkrement
Zyklisches Hochzählen von 0 bis 1000

FB300, DB300, FC2
Simulation von Trigger
Inkrement
Zyklisches Hochzählen von 0 bis 1000

FC5, DB222, SFC64
Durch DB222 (Text Bibliothek) kann die Zeichen und Texte der Stringvariable eingegeben werden. SFC64 kann die eingegebenen Inhalte von Stringvariablen zyklisch verändern. Der Zyklus kann manuell eingestellt werden.

4.2 Projektierung Einzelplatz System auf WinCC

In diesem Abschnitt wird das WinCC Projekt dargestellt. Es enthält Einrichtung des Kommunikationskanals, Anlegen der Prozessvariablen, Erstellung von Tag Logging und Alarm Logging, Konstruktion der Testbilder.

4.2.1 Erstellung eines neuen WinCC Projektes

Um das weitere Projekt einfacher von PG (Engineering Station) nach WinCC Server zu laden, sollen das WinCC Projekt in STEP7 mit SPS Projekt integriert werden. Also wird das WinCC Projekt auf der PC-Station in SIMATIC Manager erstellt.

1. Eine neue SIMATIC PC-Station einfügen -> Name „WinCCServer“ eingeben -> Konfiguration -> Hardware: „WinCC Appl“ und „IE Allgemein“ im Rack hinzufügen

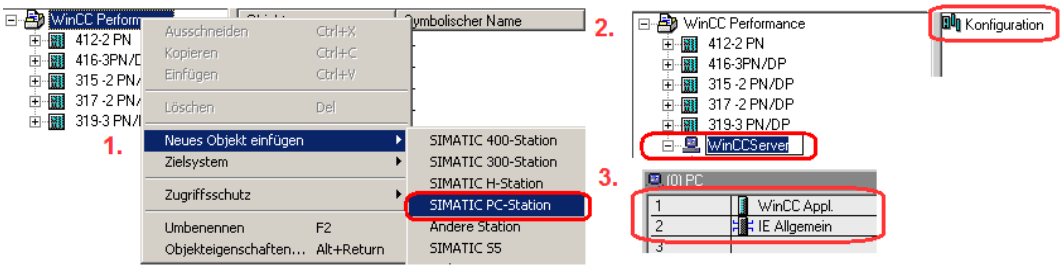


Abbildung 4-8: Hinzufügung einer neuen PC-Station

2. Kommunikation der PC-Station konfigurieren -> „Eigenschaft“ -> Ethernet(1) -> IP-Adresse und Subnetzmaske eingeben -> „OK“
IP-Adresse: 172.16.70.94
Subnetzmaske: 255.255.0.0

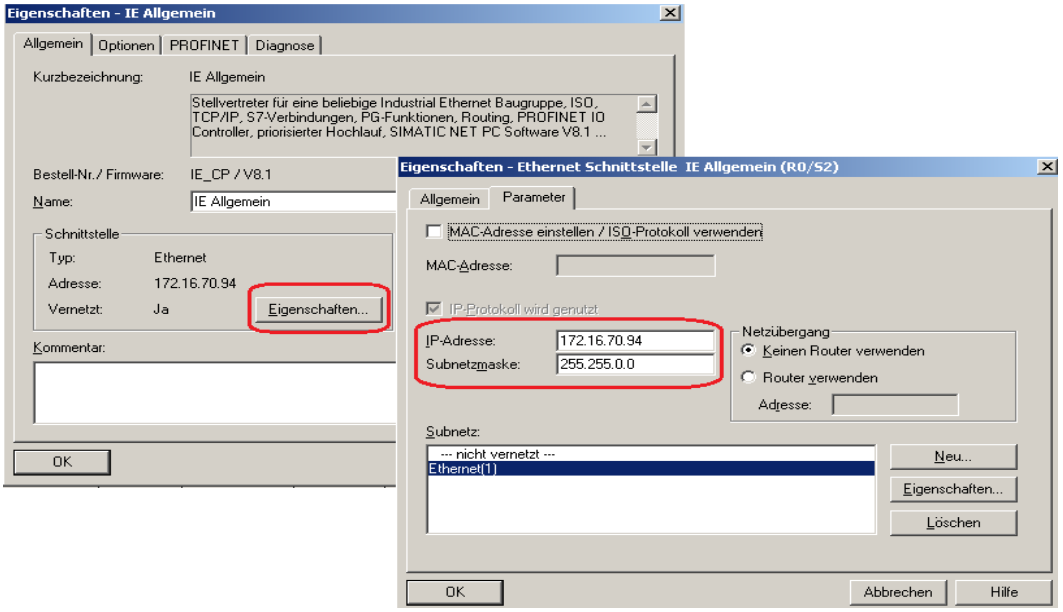


Abbildung 4-9: Konfiguration der Kommunikation von PC-Station

3. Eine Name: „WinCCPerformanceProjekt“ eingeben –> „Objekt öffnen“

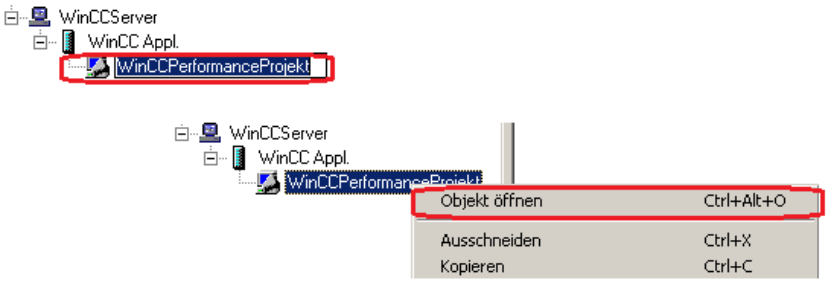


Abbildung 4-10: Öffnen WinCC Projekt

4. Neuen Kommunikationstreiber hinzufügen („Variablenhaushalt“ -> „Neuen Treiber hinzufügen“ -> „SIMATIC S7 Protocol Suite.chn“ auswählen)

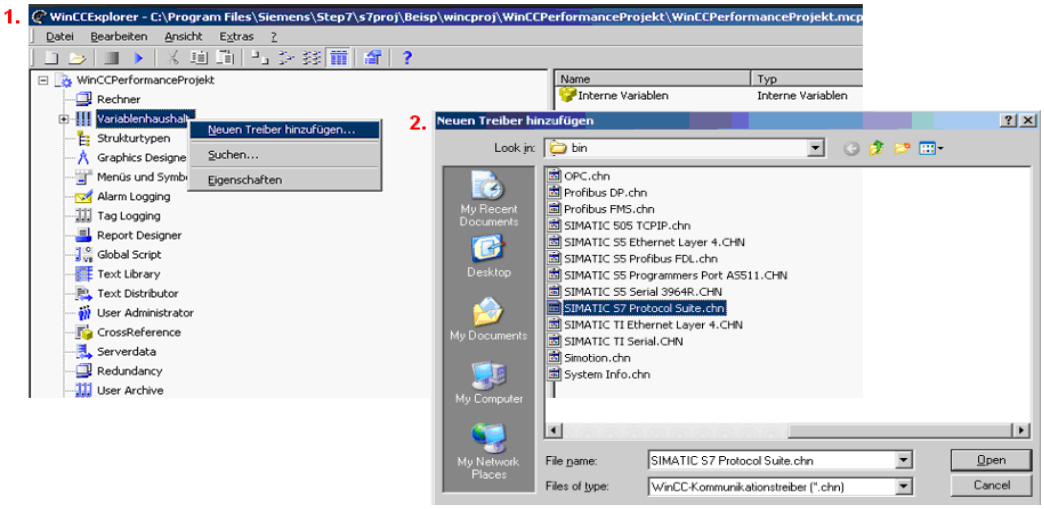


Abbildung 4-11: Hinzufügung eines Kommunikationstreibers

5. Neue Verbindung erstellen („TCP/IP“ -> „Neue Verbindung“ -> Name: „CPU317“ eingeben -> „Eigenschaften“ -> IP-Adresse, Rack-Nummer, Steckplatz-Nummer der CPU317 eingeben -> „OK“)

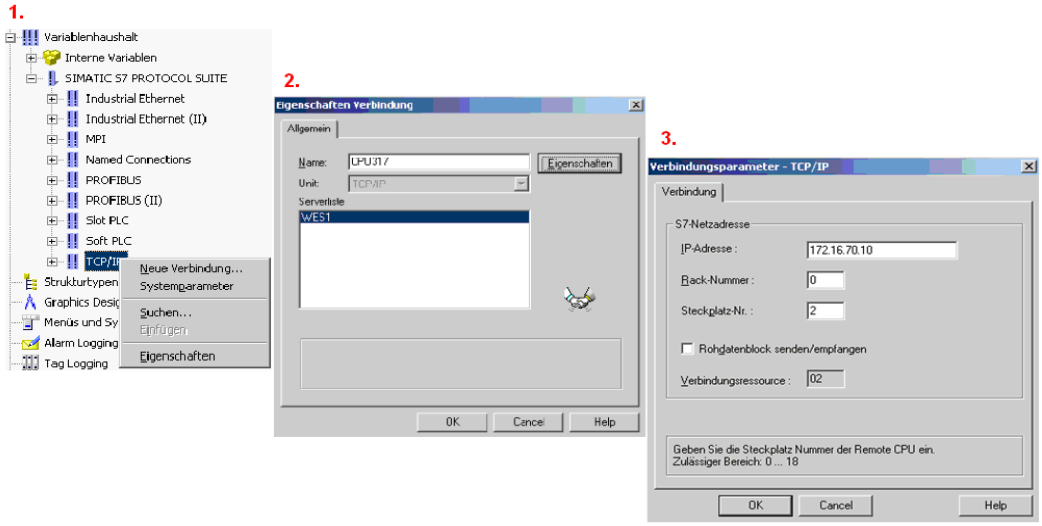


Abbildung 4-12: Erstellung einer neuen Verbindung

4.2.2 Anlegen der Variablengruppen und Variablen

Die Variablengruppen werden 5 Arten eingeteilt.

1. „Alarm Logging“ erhält 40 DWORD (Vorzeichnenloser 32-Bit Wert) Variablen zu den Alarmmeldungen von DB1-DB10.
2. „Tag Logging“ erhält insgesamt 500 DWORD Variablen zu den Archivierungen von DB11-DB20, 50 Variablen pro DB.
3. „TriggerDB21“ - „TriggerDB30“ beinhalten die verwendeten DWORD-Prozessvariablen in Testbildern (100 Variablen pro Gruppe).
4. Die 1000 String (Textvariable 8-Bit Zeichnensatz) Variablen werden in „StringDB31“ - „StringDB34“ eingefügt, 250 Variablen pro Gruppe.

1. Die Variablengruppe und Variablen von CPU 317-2PN/DP anlegen

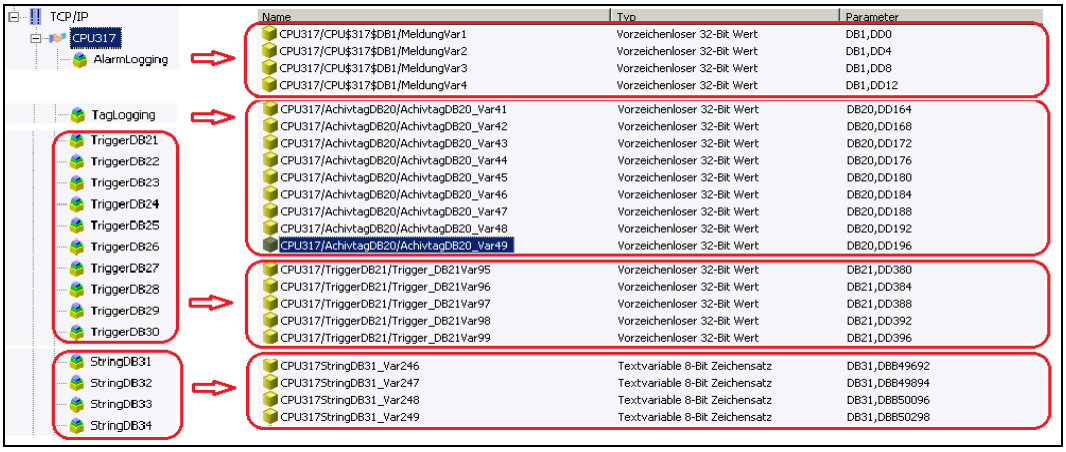


Abbildung 4-13: Übersicht der allen Datenbausteine und Anteil der zugehörigen Variablen

2. Verbindungen der anderen Steuerungen und die zugehörige Variablen hinzufügen

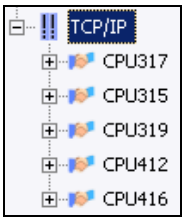


Abbildung 4-14:Verbindungen von 5 Steuerungen

Weil die Anzahl der Variablen groß ist, wird VBA benutzt, um die Variablen schnell zu hinzufügen. VBA ist ein Werkzeug von WinCC. Mit Hilfe der WinCC VBA können Objekte, Trigger, Verbindungen, Variablen automatisiert werden. VBA wird in der Grafik Designer erstellt, also das Programm wird in der pdl. Datei gespeichert. Das VBA Code siehe im WinCC Projekt.

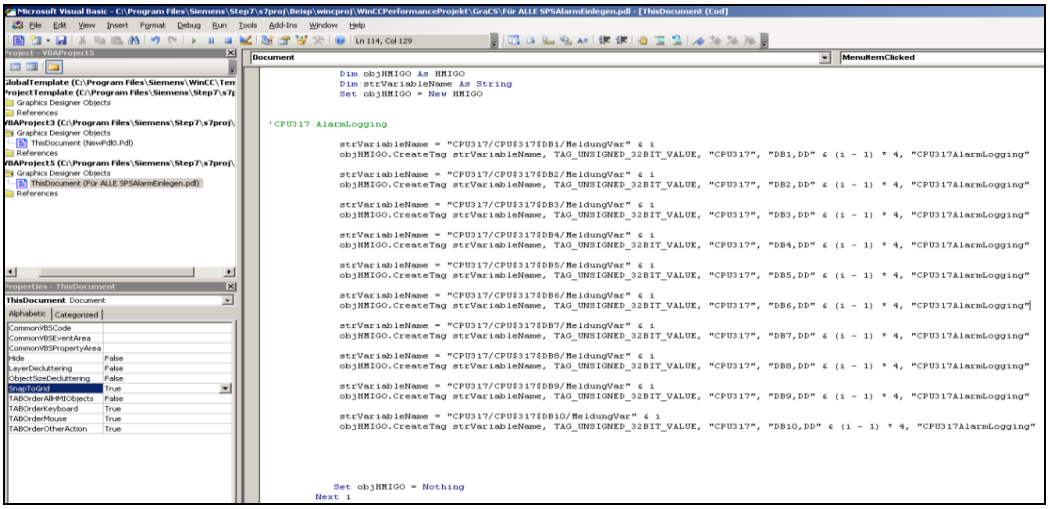


Abbildung 4-15: Ansicht des Programms in VBA

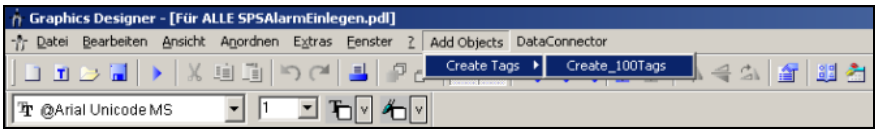
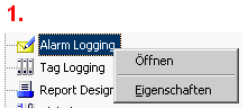


Abbildung 4-16: Hinzufügung der Variablen durch VBA

4.2.3 Erstellung von Alarm Logging

1. Bit-Meldungen anlegen („Alarm Logging“ –> „öffnen“ -> Klasse: „Alarm“ -> Art: „Alarm High“ -> Meldvariable: „CPU317/CPU317DB1/MeldungVar1“ -> Meldbit: „0“ -> WinCC Meldtext „MeldungVar1“ -> Ereignis: „AlarmMeldung1“)



2.

Nummer	Klasse	Art	Priorität	Meldevariable	Meldebit
1	Alarm	Alarm High	0	CPU317/CPU317DB1/MeldungVar1	0
2	Alarm	Alarm High	0	CPU317/CPU317DB1/MeldungVar1	1

Abbildung 4-17: Hinzufügung der Alarmmeldungen

Jede Meldvariable hat 32 Bit, also von 0 bis 31. 1Bit kann eine Meldung setzen, so gibt es 1280 Mitmeldungen pro Steuerung, insgesamt 6400 Bitmeldungen von 5 Steuerungen.

2. 6400 Bitmeldungen von 5 Steuerungen hinzufügen

	6391	Error	Alarm	0		22
	6392	Error	Alarm	0		23
	6393	Error	Alarm	0		24
	6394	Error	Alarm	0		25
	6395	Error	Alarm	0		26
	6396	Error	Alarm	0		27
	6397	Error	Alarm	0		28
	6398	Error	Alarm	0		29
	6399	Error	Alarm	0		30
	6400	Error	Alarm	0		31

Abbildung 4-18: Anteil der insgesamten Alarmmeldungen

3. in der Menüzelle klicken zum Speichern der Datei

4.2.4 Erstellung von Tag Logging

1. Archive voreinstellen („Tag Logging“ –> „öffnen“ -> Archive: „Voreinstellung“ -> Analogvariable: „Archivierung“ -> Archivierungsart: „zyklisch“ -> Erfassung: „1 second“ -> Faktor: „1“ -> Zyklus: „1 second“)

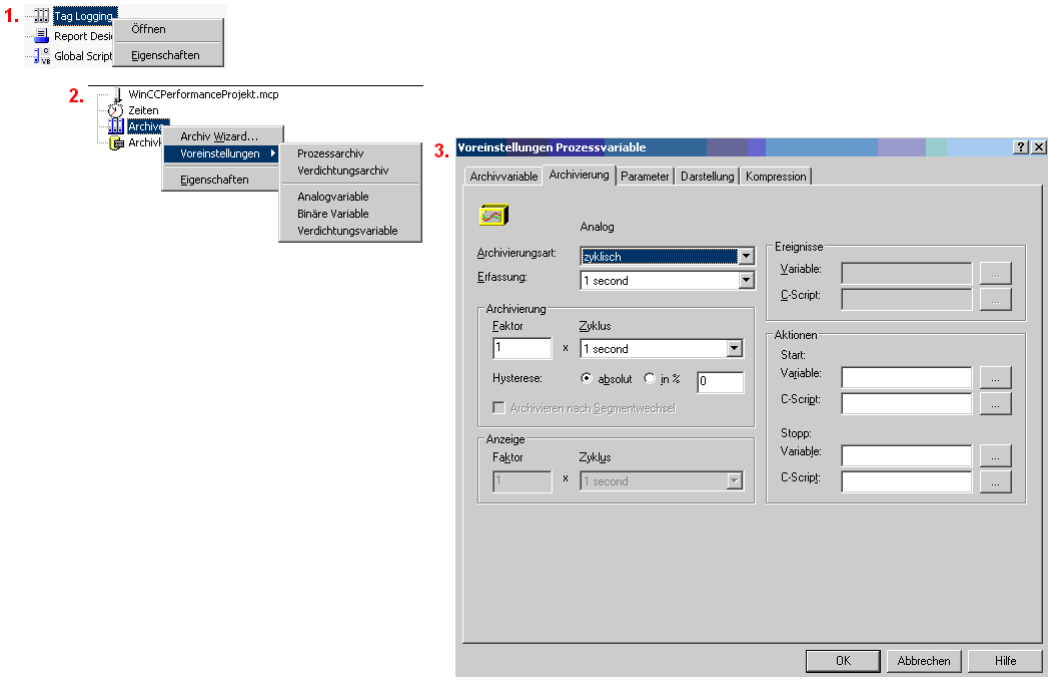


Abbildung 4-19: Voreinstellung von Tag Logging

2. Archivierungsvariablen hinzufügen (Archive -> „ArchivWizard“ -> „317ArchivtagDB11_1s“ als Archivname eingeben -> Archiv Typ: „Prozesswertarchiv“ -> Weiter -> die Variablen aus Gruppe auswählen -> Anwenden)

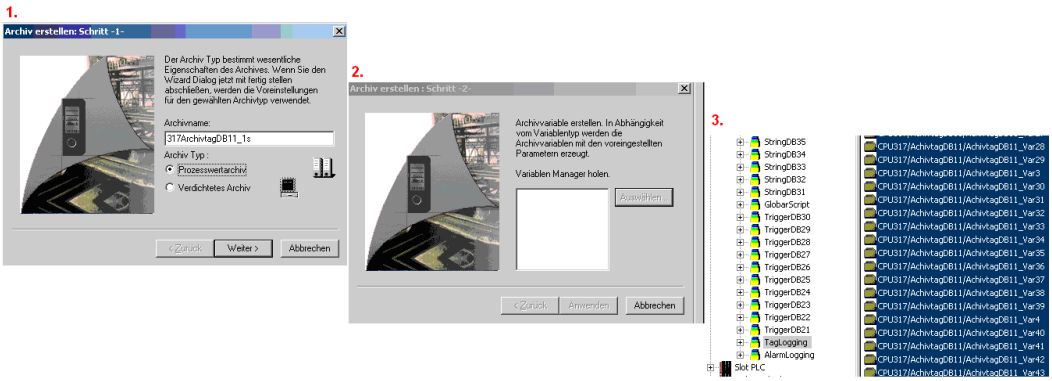


Abbildung 4-20: Hinzufügung der Archivierungsvariablen

3. Durch die gleiche Methode die andere Archivgruppe hinzufügen (ArchivtagDB11_1s bis ArchivtagDB20_1s von CPU 315, CPU317, CPU319, CPU412, CPU416)

Archivname	Archivtyp
317ArchivtagDB11_1s	Prozesswertarchiv
317ArchivtagDB12_1s	Prozesswertarchiv
317ArchivtagDB13_1s	Prozesswertarchiv
317ArchivtagDB14_1s	Prozesswertarchiv
317ArchivtagDB15_1s	Prozesswertarchiv
317ArchivtagDB16_1s	Prozesswertarchiv
317ArchivtagDB17_1s	Prozesswertarchiv
317ArchivtagDB18_1s	Prozesswertarchiv
317ArchivtagDB19_1s	Prozesswertarchiv
317ArchivtagDB20_1s	Prozesswertarchiv

Abbildung 4-21: Archivgruppen von S7-317 2PN/DP

4.2.5 Prozessbilderzeugung

Im diesem Abschnitt wird die Erzeugung der WinCC Prozessbilder dargestellt.
Bei der Erzeugung der Bilder benutzen wir:
4 Hauptobjekte: EA_Feld; Rechteck; AlarmControl; OnlineTableControl.
4 Dynamisierungen: C-Aktion, VBS-Aktion, Dynamik-Dialog, direkte Variablenanbindung
2 Trigger: Zyklischer Trigger, Variablentrigger
2 Animationen: EA_Feld zeigt die Werte der Prozessvariablen an; Rechteck mit der Animation: Bei der Wert der verbundenen Variable großer als 500 wird die Hintergrundfarbe grün, bei anderem Status rot.

<u>Erstellung der Kontrollbilder</u>

1. Erstellung vom Bild: „Tagloggging“ („WinCC OnlineTableControl“ -> Fenster im Bild einlegen-> Wertspalten im Dialogfeld hinzufügen -> „OK“)

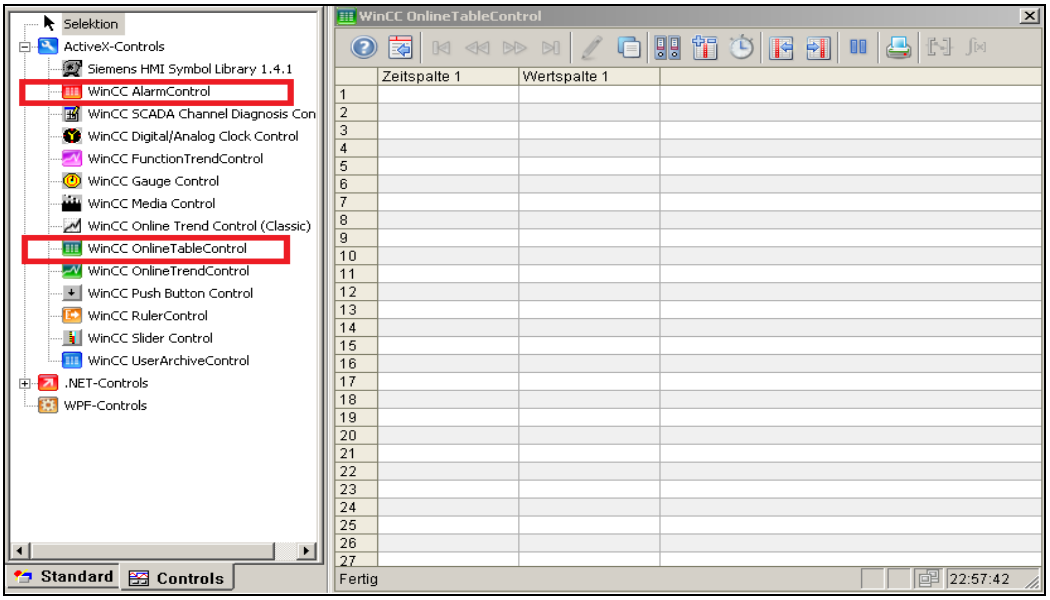


Abbildung 4-22: Erstellung des Kontrollbildes „Taglogging“

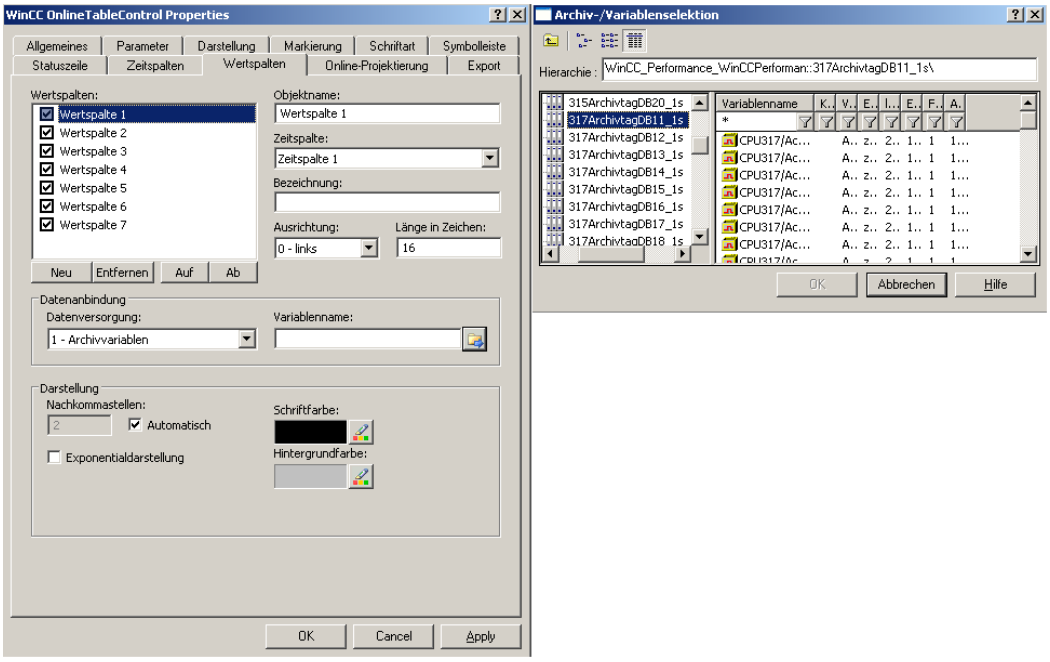


Abbildung 4-23: Hinzufügung der Wertspalten

2. Erstellung vom Bild „Alarm Logging“ („WinCC AlarmControl“ -> Fenster im Bild einlegen-> „OK“)

Hinweis: Der Kontrollbild dient nur für die Simulation einer wirklichen Industrieumgebung erforderlich, denn es hat keine Auswirkung auf der Messergebnisse.

<u>Erstellung der Testbilder</u>

Rechteck mit Dynamisierungsart: C-Aktion; Triggerart: Variablentrigger

1. Ein Rechteck im Bild hinzufügen -> „Objekteigenschaft“ -> „Farbe“ –> „Hintergrundfarbe“ -> „Dynamik C“ -> das Programm im Programmfeld eingeben -> „OK“

2. Trigger klicken -> „Ereignis“ -> „Variable“

3. Selektionsbutton klicken -> die Variable auswählen -> „OK“

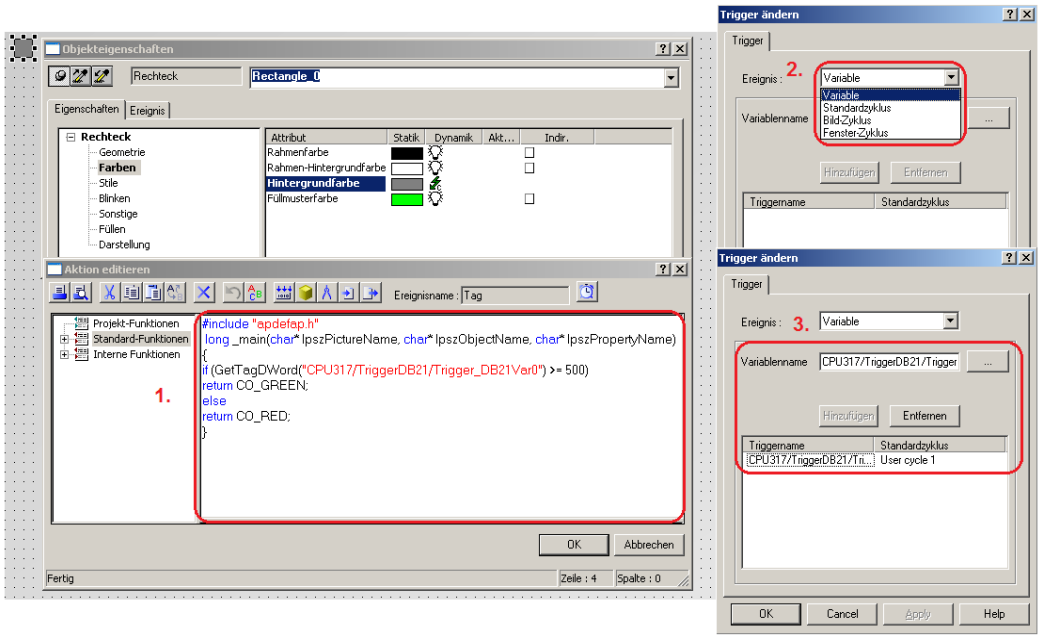


Abbildung 4-24: Erzeugung des Rechtecks mit C-Aktion, Variablentrigger

Die Konfiguration vom zyklischen Trigger ist ähnlich wie die bei Variablentrigger.

1. Trigger klicken -> „Ereignis“ -> „Standartzyklus“

2. Name vom Trigger eingeben -> die Zykluszeit im Feld „Zyklus“ auswählen (hier ist „User cycle 1[8]“) -> „OK“

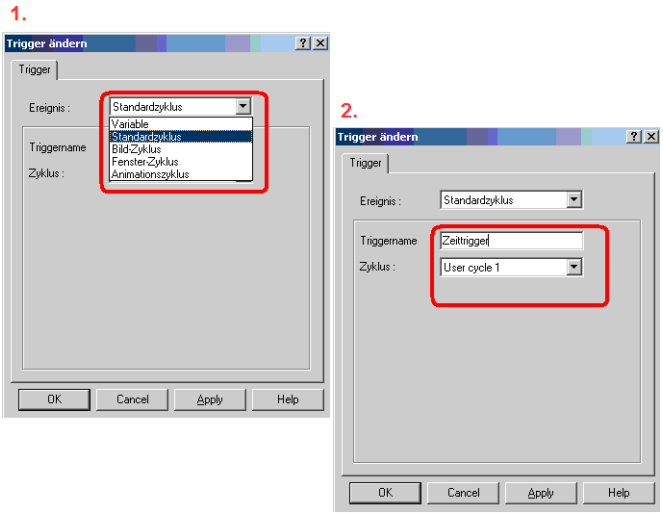


Abbildung 4-25: Einstellung vom zyklischen Trigger bei C-Aktion und VBS-Aktion

Die Konfiguration des Rechtecks bei VBS-Aktion mit dem zyklischen Trigger od. Variablentrigger ist gleich wie die bei C-Aktion. Hier wird es nicht dargestellt. (siehe Ab.4-24 und Ab.4-25)

[8] Wenn „User cycle“ ausgewählt wird, kann der Aktualisierungszyklus manuell eingestellt wird. Es ist flexibel für die Einstellung der Umgebung von weiteren Messaufgaben.

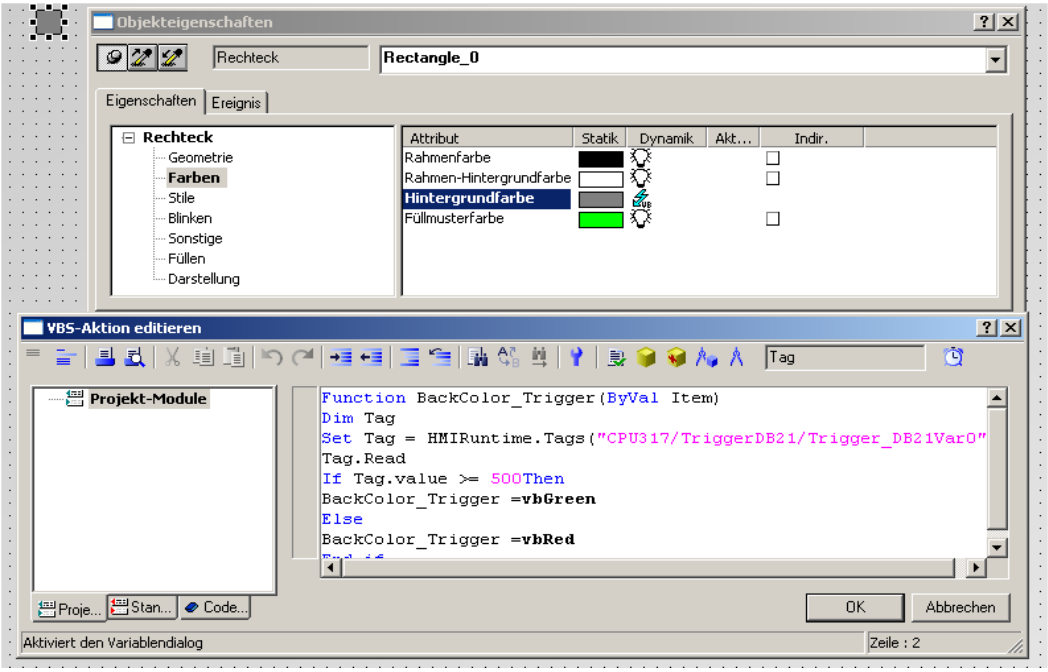


Abbildung 4-26: Erzeugung des Rechtecks mit VBS-Aktion

Rechteck mit Dynamisierungsart: Dynamik-Dialog

Ein Rechteck im Bild hinzufügen -> „Objekteigenschaft“ -> „Farbe“ –> „Hintergrundfarbe“ -> „Dynamik-Dialog“

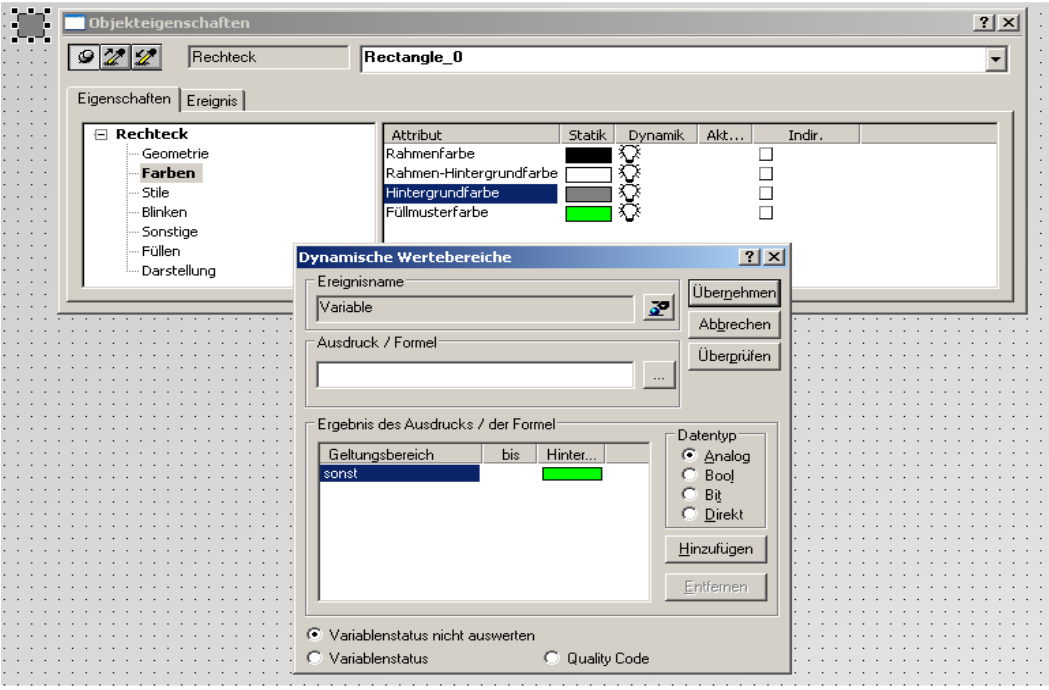


Abbildung 4-27: Erzeugung des Rechtecks mit Dynamik-Dialog

Erzeugung der Animation:

1. „Analog“ als Datentyp auswählen -> einen neuen Wertebereich hinzufügen -> Werteingabe: bis 500 –> Hindergrundfarbe auswählen -> „OK“

2. Button rechts von „Ausdruck / Formel“ klicken –> „Variable“ (Variable auswählen)

3. Trigger klicken (Die Methode der Hinzufügung vom Trigger ist gleich wie die bei C - und VBS - Aktion)

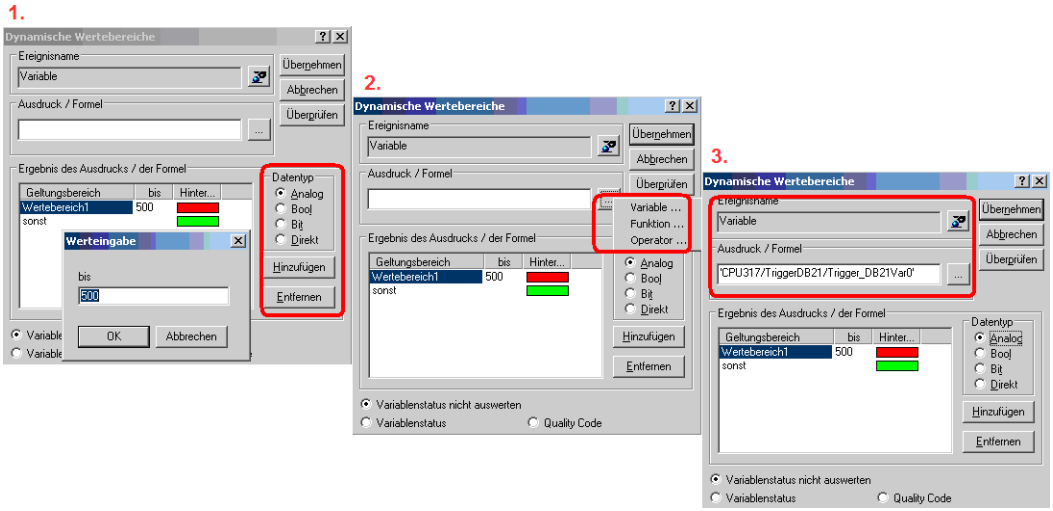


Abbildung 4-28. Einstellung bei Dynamisierungsart: Dynamik-Dialog

EA_Feld mit Dynamisierungsart: direkte Variablenanbindung

Ein EA_Feld ins Bild einfügen -> „Konfigurationsdialog“ -> Variable auswählen –> Aktualisierungszeit eingeben (hier ist „Unser cycle 1“)

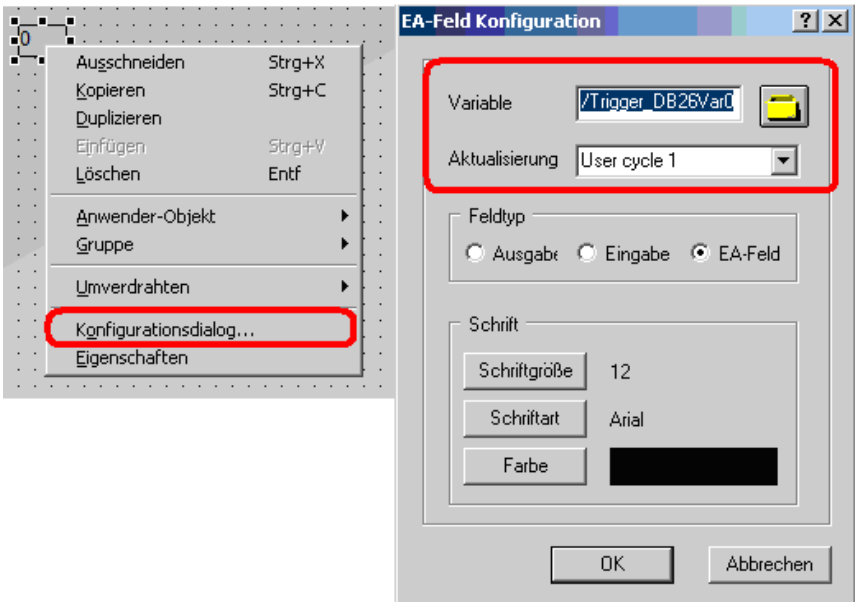


Abbildung 4-29: Erzeugung vom EA_Feld mit der direkten Variablenanbindung

Hinweis: Für die richtige Darstellung des Variablenwerts im RT muss abhängig von Datentyp des Datenformats gewählt werden.

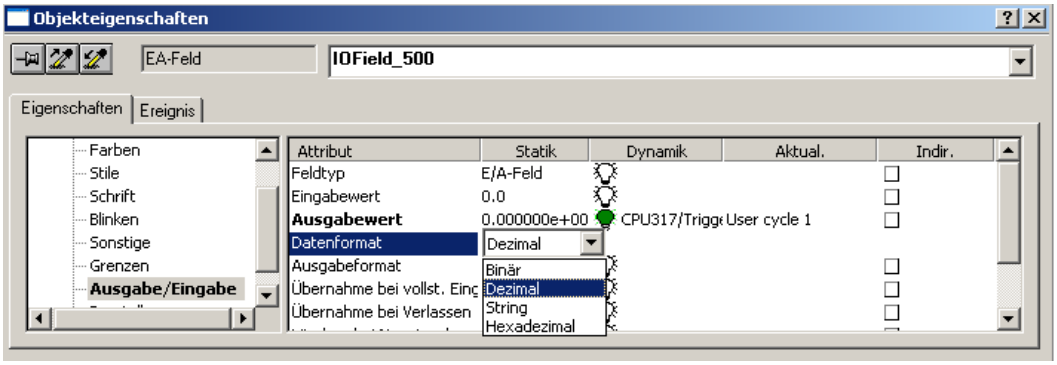


Abbildung 4-30: Einstellung des Datenformats vom EA_Feld

Die nachfolgenden Tabellen zeigen die Eigenschaften der Objekte und Testbilder an. Die graphische Darstellung siehe A [02] im Anhang.

Tabelle 4-4: Eigenschaften der Objekte

Objekt ID	Objekttyp	Datentyp von Prozessvariablen	Triggerart	Dynamisierungsart
Obj.1	EA_Feld	String	Variablentrigger	direkte Variablenanbindung
Obj.2	EA_Feld	DWORD	Variablentrigger	direkte Variablenanbindung
Obj.3	Rechteck	DWORD	Zyklische Trigger	C-Aktion
Obj.4	Rechteck	DWORD	Zyklische Trigger	VBS-Aktion
Obj.5	Rechteck	DWORD	Zyklische Trigger	Dynamik-Dialog
Obj.6	Rechteck	DWORD	Variablentrigger	C-Aktion
Obj.7	Rechteck	DWORD	Variablentrigger	VBS-Aktion
Obj.8	Rechteck	DWORD	Variablentrigger	Dynamik-Dialog

Kontrollbilder

Tabelle 4-5: Kontrollbilder: „Tag Logging“ und „Alarm Logging“

Bildnummer	Bildname	Objekt
KB1	„Tag Logging“	WinCC Online Table Control
KB2	„Alarm Logging“	WinCC Alarm Control

Testbilder

Tabelle 4-6: Testbilder zum Test: „Einstellung des Lesedienstes“

Bildnummer	Bildname	Objekt ID	Anzahl	Variable
TB1	„Lesedienst“	Obj.1	1	Var.0 von DB31

Tabelle 4-7: Testbilder zum Test: „Einstellung der Steuerung“

Bildnummer	Bildname	Objekt ID	Anzahl	Variable
TB2	„Belastungstest“	Obj.1	500	Variablen von DB31-DB32
TB3	„Belastungstest 4DB“	Obj.1	1000	Variablen von DB31-DB34

Tabelle 4-8: Testbilder zum Test: „Auswirkung aus Bildaufbau und Konfiguration von WinCC“

Bildnummer	Bildname[9]	Objekt ID	Anzahl	Variable	Triggerart[10]
TB4	„Zeittrigger mit C Rechteck“	Obj.3	500	DB21-DB25	rein
TB5	„Zeittrigger mit VBS Rechteck“	Obj.4	500	DB21-DB25	rein
TB6	„Zeittrigger mit DynamikDialog Rechteck“	Obj.5	500	DB21-DB25	rein
TB7	„Variablentrigger mit C Rechteck“	Obj.6	500	DB21-DB25	rein
TB8	„Variablentrigger mit VBS Rechteck“	Obj.7	500	DB26-DB30	rein
TB9	„Variablentrigger mit DynamikDialog Rechteck“	Obj.8	500	DB26-DB30	rein
TB10	„Zeittrigger mit C“	Obj.3	500	DB21-DB25	gemischt
		Obj.2	500	DB26-DB30	
TB11	„Zeittrigger mit VBS“	Obj.4	500	DB21-DB25	gemischt
		Obj.2	500	DB26-DB30	
TB12	„Zeittrigger mit DynamikDialog“	Obj.5	500	DB21-DB25	gemischt
		Obj.2	500	DB26-DB30	
TB13	„Variablentrigger mit C“	Obj.6	500	DB21-DB25	rein
		Obj.2	500	DB26-DB30	
TB14	„Variablentrigger mit VBS“	Obj.7	500	DB21-DB25	rein
		Obj.2	500	DB26-DB30	
TB15	„Variablentrigger mit DynamikDialog“	Obj.8	500	DB21-DB25	rein
		Obj.2	500	DB26-DB30	
TB16	„Zeittrigger mit C 500“	Obj.3	250	DB21-DB25	gemischt
		Obj.2	250	DB26-DB30	

[9] Bildname ist die Name der pdl. Datei im WinCC Projekt

[10] Die Bedeutung hier ist, ob die Objekte im Testbild mit nur 1 Art oder 2 verschiedenen Arten von Trigger projektiert werden?

4.3 Erweiterung des WinCC Projektes

In diesem Abschnitt wird die Erweiterung des WinCC Projektes dargestellt, um die nachfolgende Messaufgaben reibungslos auszuführen.

4.3.1 Konfiguration der Verbindung mit PROFIBUS auf WinCC

1. Eine neue Verbindung mit PROFIBUS hinzufügen („Neue Verbindung“-> „CPU_317_PB“ als die Name eingeben -> Netzadresse eingeben -> „OK“)

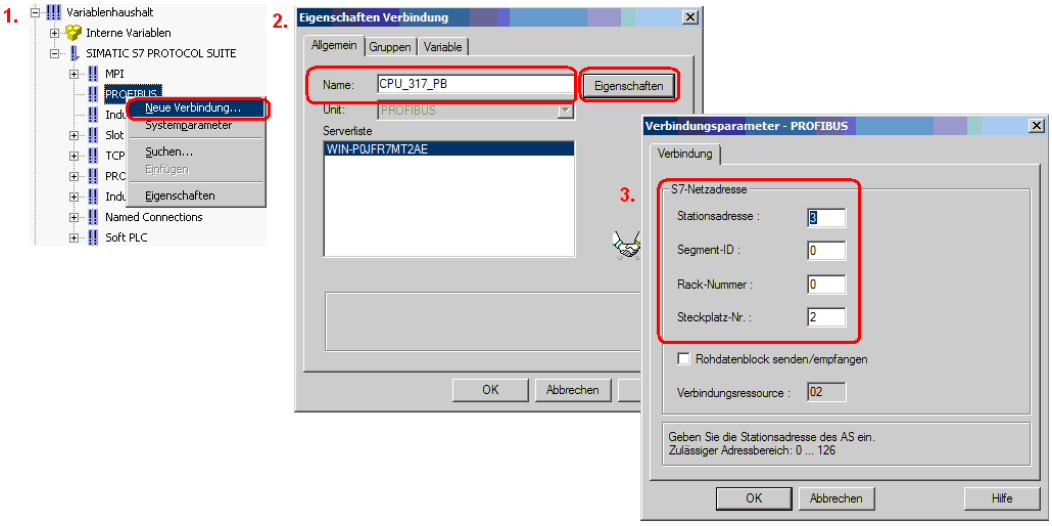


Abbildung 4-31: Konfiguration der Verbindung mit PROFIBUS

Tabelle 4-9: Datei der Adresse von CPU 317-2PN/DP und CPU 412-2PN

	Stationadresse	Segment-ID	Rack.Nr	Steckplatz-Nr
CPU 317 2PN7DP	3	0	0	2
CPU 412 2PN	2	0	0	3

2. Die Variablen von Verbindung TCP/IP ausschneiden -> in den neuen Verbindungen einfügen

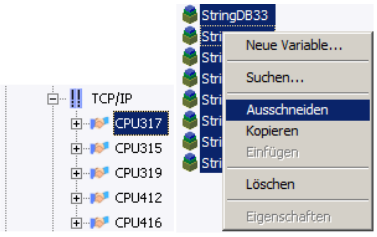


Abbildung 4-32: Hinzufügung der Variablen für Verbindung mit PROFIBUS

3. Netzkarte für PROFIBUS konfigurieren („Systemparameter“-> Netzkarte: „CP5623.PROFIBUS.1“ auswählen-> „OK“)

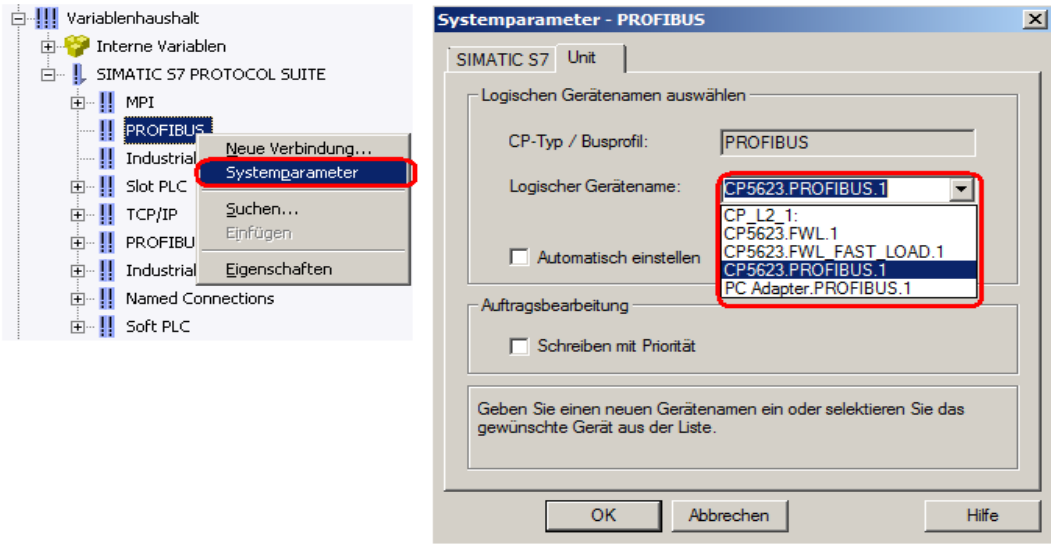


Abbildung 4-33: Konfiguration der Netzkarte für PROFIBUS

4.3.2 Konfiguration von WinCC Client

1. Eine WinCC Client Station im WinCC Server Projekt erstellen („Neuer Rechner“ -> die Name: „WES1“ von WinCC Projekt auf ES eingeben-> Rechnertyp: WinCC Client)

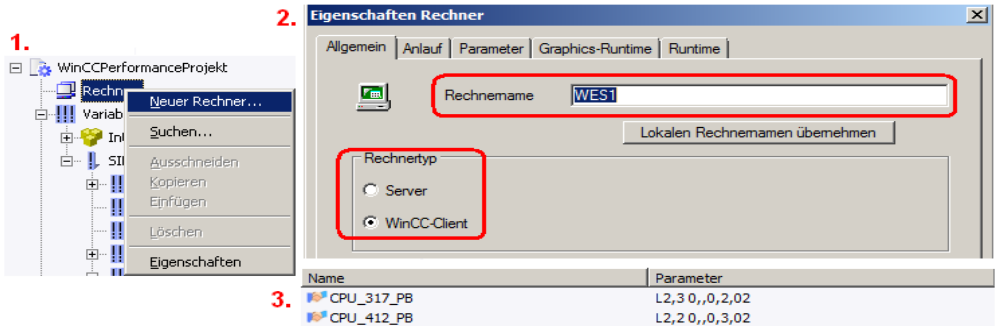


Abbildung 4-34: Erstellung eines WinCC Clients

2. Konfiguration von WinCC Client („User Adminstrator“-> eine neue dient „client“ hinzufügen-> Wählen Nr.1, Nr.1000, Nr.1001 aus, um die Berechtigung einzustellen -> Trigger in der Menüzelle klicken -> Passwort aufstellen -> „“)

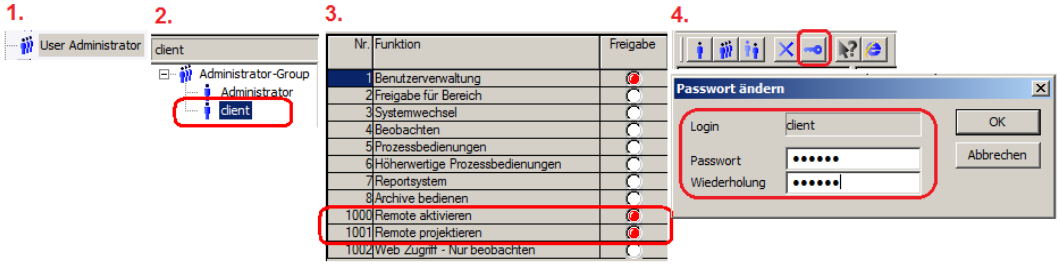


Abbildung 4-35: Konfiguration der Berechtigung von WinCC Client

3. Serverdata auf der ES erzeugen („OS-Server zuordnen“-> „OK“) Danach wird WinCC Projekt automatisch geöffnet und Serverdata (WinCC_Performance) erzeugt.

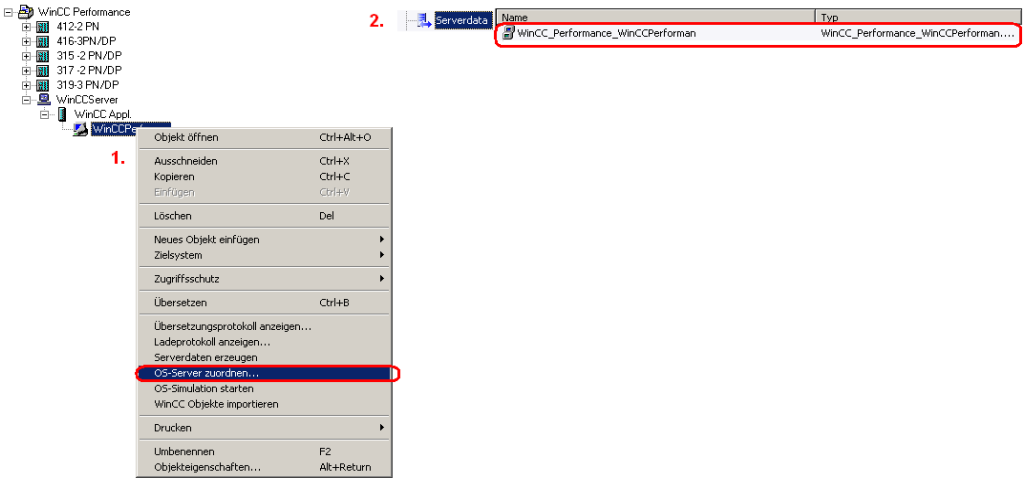


Abbildung 4-36: Erzeugen der Serverdata auf ES

4. Auf der WinCC Client Station (ES) (das Verzeichnis „Simatic Shell“ öffnen -> Objekt: „WinCCPerformanceProjekt“ -> „Interconnect“ -> Login Name und Password eingeben -> „OK“)

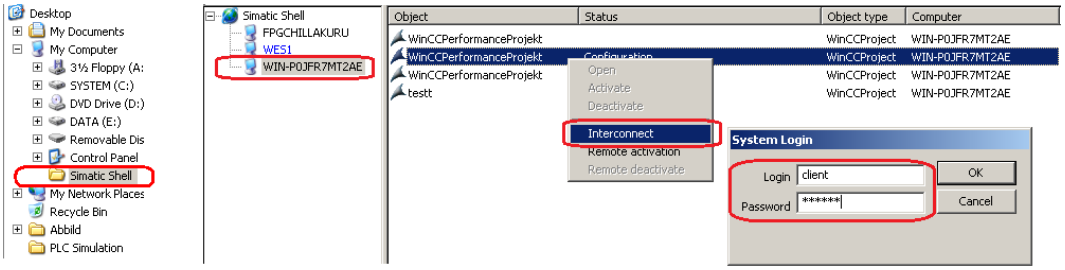


Abbildung 4-37: Starten des WinCC Clients

Dann kann die ES als WinCC Client verwendet werden. Wenn die Runtime am Server und Client aktiviert sind, kann von der ES (WinCC Client) aus die Prozessbilder vom WinCC-Server beobachtet und bedient werden.

5 Performancewerte messen und ermitteln

5.1 Messtool

Durch die Beobachtung der Verhältnisse und Verlauf von den Telegrammen zwischen WinCC und Steuerungen werden die Ergebnisse der Kommunikationsperformance und die Leistungsfähigkeit von WinCC bei der Runtime erfasst. Beim Messen mit Ethernet (Protokolle: TCP/IP) wird eine günstige Messtool: Wireshark benutzt. Als ein Werkzeug zur Analyse der Kommunikationsparameter, Wireshark liefert viele Möglichkeiten, um die Telegramme zu erfassen und zu analysieren. Der Vorteil von Wireshark ist, dass es die Telegramme die genaue Informationen der 7 Schichte des ISO-OSI-Referenzmodells[11] zwischen Kommunikationspartner zeigen kann.

<u>Einstellungen von Wireshark für die Vorbereitung zum Messen</u>

1. „Wireshark“ öffnen -> Capture Operation -> Netzkarte auswählen (also die Port der Kommunikationspartner). Bei diesem Test wird der Redundancy_Bus: Intel(R) 82579LM ausgewählt.

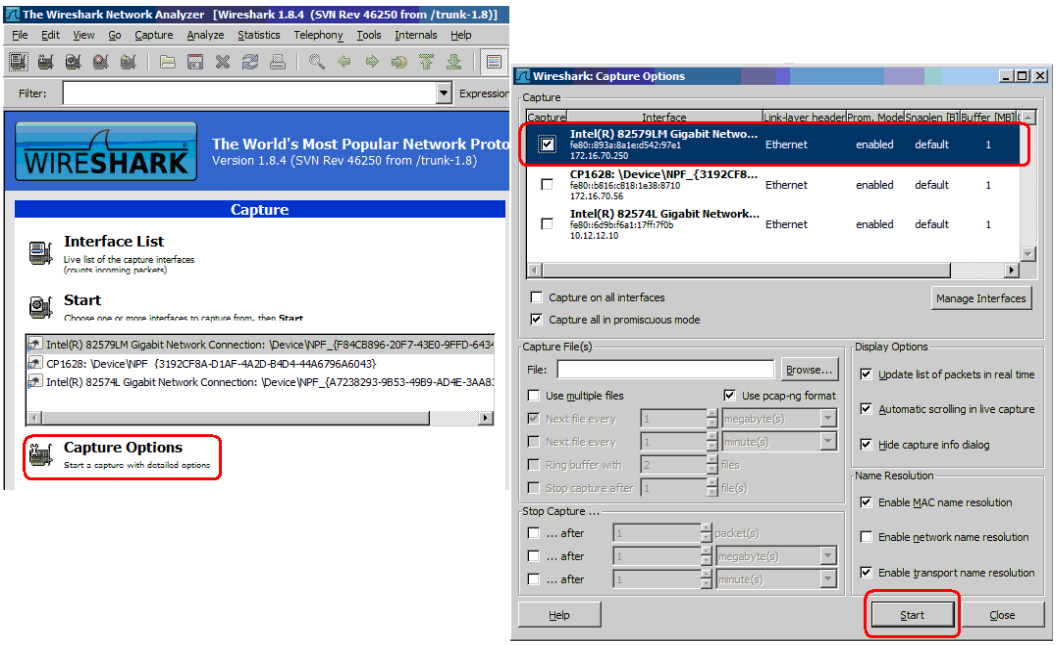


Abbildung 5-1: Konfiguration der Wireshark

Hinweis: WinCC Kommunikationsmodul ist CP 1628 bei TCP/IP. Beim Test müssen alle Kommunikationspartner mit SWITCH verbunden werden. Der Port mit Redundancy_Bus ist die Spiegel von Multi – Broadcast. Durch den Port werden alle Telegramme zwischen Kommunikationspartner von Wireshark ermittelt.

[11] Die genaue Darstellung siehe 2.1.5 vom Siemens Handbuch WinCC Communication Manua Band 1

2. „Start“ klicken, der Test beginnt

Danach werden die Telegramme (Trace) aufgezeichnet (siehe Abbildung unter).

Von den aufgezeichneten Telegramme sind nur solche für unseren Test relevant, welche die Inhalte von Prozessvariablen (z.B. der Anforderung von WinCC od. die Antwort von SPS mit der Information über die Prozessvariablen) enthalten.

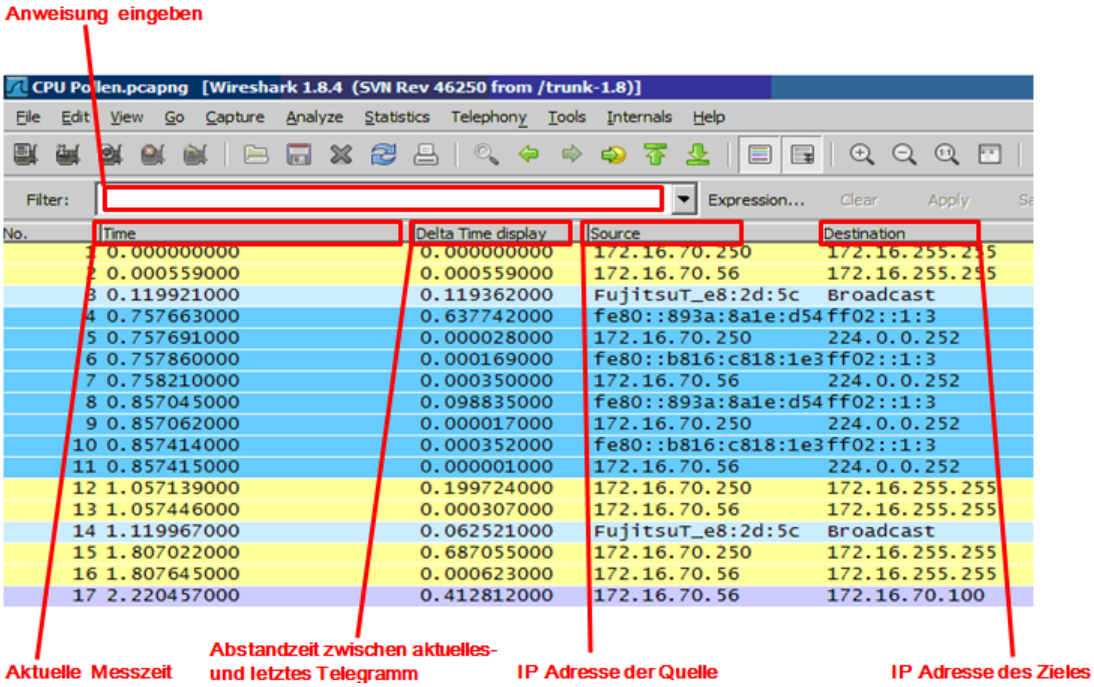


Abbildung 5-2: Testfeld von Wireshark

3. Geben die Anweisung „tcp.stream&&t125“ oder „t125“ in der Zelle „Filter“ ein, um die gewünschten Telegramme, sowie die Telegramme zwischen WinCC Server und Steuerungen zu filtern. Die Anweisung kann gespeichert werden.

In der Applikation kann man folgende Information des Telegramms sehen: die ID Nummer (No.), die Uhrzeit (Time), die Abstandzeit (Delt Time display) zwischen aktuellen Telegramm und letzten Telegramm, Quelle IP Aderesse (Source), Ziel IP Aderesse (Destination), Protocol, Länge (Length) und die andere Information.

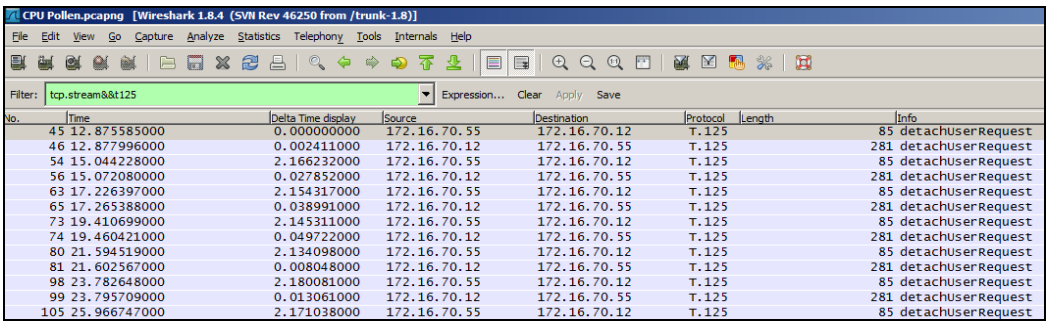


Abbildung 5-3: Telegramm mit dem Protokoll: TCP/IP

5.2 Messaufgaben

5.2.1 Einstellung des Lesedienstes

Einflussfaktor:

- Lesedienstart[12]

Im Abschnitt 2.1.2 wird die Struktur von WinCC Datenmanager dargestellt. Der Datenmanager von WinCC hat die Aufgabe, die verschiedenen Applikationen des WinCC Projekts mit Prozessdaten zu versorgen. Dazu muss dieser seinen Datenbestand in irgendeiner Form in den geforderten Zyklen aktualisieren. Die Art und Weise, wie er dies durchführt, hat natürlich Einfluss auf die Belastung des Kommunikationssystems.
Bei „Pollen“ werden Daten aus der AS nach Anforderung von WinCC einmaliges, sporadisches gelesen. Es werden für jede Aktualisierung zwei Kommunikationstelegramme benötigt. Die WinCC Station sendet an das Automatisierungsgerät eine Anforderung, welche von diesem über ein Antworttelegramm (oder mehrere Antworttelegramme) bearbeitet wird.
Bei „Zykluslesen“ sendet WinCC an die CPU einmalig einen Auftrag, um in der CPU einen zyklischen Leseauftrag einzurichten (z.B. bei Bildanwahl, dabei wird auch der Lesezyklus vorgegeben, z.B. 500 ms). Ist dies erfolgt, sendet die CPU selbstständig die Lese-Daten während der Lebensdauer des Auftrags im vorgegebenen Zyklus, z.B. 500 ms, an WinCC. WinCC beendet den zyklischen Leseauftrag (z.B. bei Bildabwahl).
Bei „Zykluslesen bei Änderung“ richtet WinCC einen Auftrag für das änderungsgesteuerte Lesen in der CPU ein. (z.B. bei Bildanwahl) Die CPU sendet während der Lebensdauer des Auftrags immer dann die angeforderten Daten an WinCC, wenn sich der Wert einer oder mehrerer der im Auftrag spezifizierten Variablen geändert hat.
In diesem Messen wird der Einfluss zur Kommunikation aus den obigen genannten 3 Lesedienstarten ermittelt.

Messmethode:

Durch Wireshark greifen wir die Leseaufträgen von WinCC und die Antworten von SPS bei der Testbild - Runtime (mit einer konstanten Prozessvariablen) im Messintervall mit den verschiedenen Lesedienstarten.

Konfiguration:

Im Dialogfeld „Systemsparameter - Industrial Ethernet“ kann man Lesedienstart einrichten.
„durch AS“ ankreuzen, um „Pollen“ zu aktivieren
„durch AS“ ankreuzen und „mit Änderungsübertragung“ nicht ankreuzen, um „Zykluslesen“ zu aktivieren
„durch AS“ ankreuzen, dabei „mit Änderungsübertragung“ ankreuzen, um „Zykluslesen bei Änderung“ zu aktivieren

[12] Die Inhalte dieses Punkts wurden von Siemens Handbuch WinCC Communication Manua Band 1: 3.1.2.2 Seite 3-10 zitiert.

Hinweis: Im Dialogfeld von WinCC können Zykluslesen und Zykluslesen bei Änderung ausgewählt, aber die wirkliche Funktionalität von Lesedienst ist abhängig von der CPU.

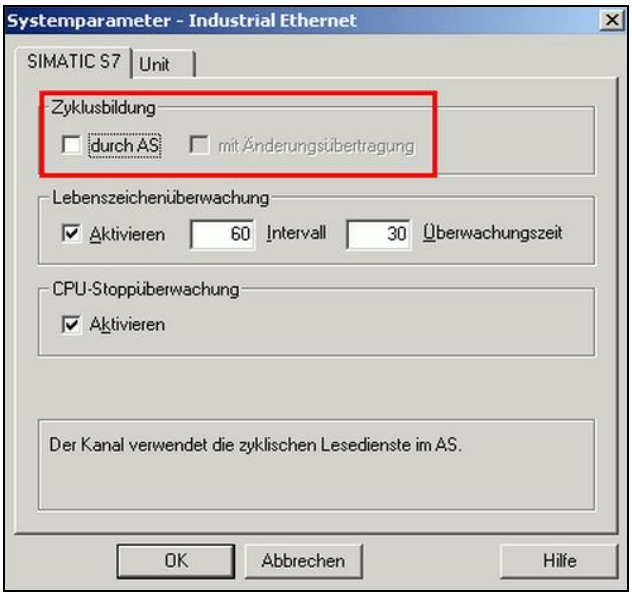


Abbildung 5-4: Einrichtung des Lesedienst s

Messumgebung:

Die Messumgebung wird die Form als Tabelle angezeigt. Die Texte mit fetter Schrift sind die Messvariablen od. Einflussfaktoren in der aktuellen Messaufgabe.

Tabelle 5-1: Einstellung der Messumgebung zum Test „Einstellung des Lesendienstes“[13]

Steuerung	OB1 Zyklus	Priorisierte BuB-Kommunikation	Zyklusbelastung durch Kommunikation	
	50ms	deaktiv	20%	
WinCC	Verbindungsart	**Lesedienstart**	Testbild	Aktualisierungszyklus der Objekt
	TCP/IP	**Pollen, Zykluslesen, Zykluslesen bei Änderung**	**TB1**	2000ms
Wireshark	Messintervall	Anweisung zum Greifen der Telegramme		
	20sec	**t125**		

Messergebnis:

Die nachfolgenden Abbildungen zeigen die Telegramme zwischen WinCC und Steuerungen bei der Testbild - Runtime im Messintervall mit verschiedenen Lesediensten. Die Tabellen fassen die Anzahl der Telegramme beim Messen zur verschiedenen CPUs.

[13] Die Einrichtungen der Steuerung werden im nächsten Abschnitt ausführlich dargestellt.

No.	Time	Delta Time display	Source	Destination	Protocol	Length	Info
252	*REF*	*REF*	172.16.70.55	172.16.70.10	T.125	85	detachUserRequest
253	0.022733000	0.022733000	172.16.70.10	172.16.70.55	T.125	281	detachUserRequest
258	2.184242000	2.161509000	172.16.70.55	172.16.70.10	T.125	85	detachUserRequest
259	2.216865000	0.032623000	172.16.70.10	172.16.70.55	T.125	281	detachUserRequest
265	4.367301000	2.150436000	172.16.70.55	172.16.70.10	T.125	85	detachUserRequest
267	4.409943000	0.042642000	172.16.70.10	172.16.70.55	T.125	281	detachUserRequest
271	6.552362000	2.142419000	172.16.70.55	172.16.70.10	T.125	85	detachUserRequest
273	6.603106000	0.050744000	172.16.70.10	172.16.70.55	T.125	281	detachUserRequest
279	8.735515000	2.132409000	172.16.70.55	172.16.70.10	T.125	85	detachUserRequest
280	8.745280000	0.009765000	172.16.70.10	172.16.70.55	T.125	281	detachUserRequest
284	10.922500000	2.177220000	172.16.70.55	172.16.70.10	T.125	85	detachUserRequest
285	10.938421000	0.015921000	172.16.70.10	172.16.70.55	T.125	281	detachUserRequest
290	13.104739000	2.166318000	172.16.70.55	172.16.70.10	T.125	85	detachUserRequest
291	13.131590000	0.026851000	172.16.70.10	172.16.70.55	T.125	281	detachUserRequest
297	15.290807000	2.159217000	172.16.70.55	172.16.70.10	T.125	85	detachUserRequest
298	15.324705000	0.033898000	172.16.70.10	172.16.70.55	T.125	281	detachUserRequest
302	17.469857000	2.145152000	172.16.70.55	172.16.70.10	T.125	85	detachUserRequest
304	17.517977000	0.048120000	172.16.70.10	172.16.70.55	T.125	281	detachUserRequest
310	19.653952000	2.135975000	172.16.70.55	172.16.70.10	T.125	85	detachUserRequest
311	19.661971000	0.008019000	172.16.70.10	172.16.70.55	T.125	281	detachUserRequest

Abbildung 5-5: Telegramme zwischen WinCC und S7-317 2PN/DP bei „Pollen“

Tabelle 5-2: Telegrammanzahl bei „Pollen“

Steuerung / **Telegramm**	319 3PN /DP	317 2PN /DP	315 2PN /DP	315 2PN/DP mit CP 343-1	416 3PN/DP	412 2PN	412 2PN mit CP 443-1
Telegrammanzahl (Summe)	20	20	20	20	20	20	20
Telegrammanzahl (WinCC)	10	10	10	10	10	10	10
Telegrammanzahl (SPS)	10	10	10	10	10	10	10

Bei „Pollen“ sendet WinCC in jedem Zyklus einen Auftrag an die Steuerung, dabei sendet die Steuerung ebenso in einem Zyklus eine Antwort zurück. Also gibt es immer 2 Telegramme zwischen WinCC und Steuerung in jedem Zyklus.

No.	Time	Delta Time display	Source	Destination	Protocol	Length	Info
247	*REF*	*REF*	172.16.70.55	172.16.70.10	T.125	99	detachUserRequest
249	0.088731000	0.088731000	172.16.70.10	172.16.70.55	T.125	295	detachUserRequest
259	2.129699000	2.040968000	172.16.70.10	172.16.70.55	T.125	295	detachUserRequest
265	4.119909000	1.990210000	172.16.70.10	172.16.70.55	T.125	295	detachUserRequest
269	6.109055000	1.989146000	172.16.70.10	172.16.70.55	T.125	295	detachUserRequest
275	8.099274000	1.990219000	172.16.70.10	172.16.70.55	T.125	295	detachUserRequest
279	10.088303000	1.989029000	172.16.70.10	172.16.70.55	T.125	295	detachUserRequest
283	12.128417000	2.040114000	172.16.70.10	172.16.70.55	T.125	295	detachUserRequest
289	14.117540000	1.989123000	172.16.70.10	172.16.70.55	T.125	295	detachUserRequest
293	16.106700000	1.989160000	172.16.70.10	172.16.70.55	T.125	295	detachUserRequest
299	18.095923000	1.989223000	172.16.70.10	172.16.70.55	T.125	295	detachUserRequest
303	20.135947000	2.040024000	172.16.70.10	172.16.70.55	T.125	295	detachUserRequest

Abbildung 5-6: Telegramme zwischen WinCC und S7-317 2PN/DP bei „Zykluslesen“

Tabelle 5-3: Telegrammanzahl bei „Zykluslesen“

Steuerung / **Telegramm**	319 3PN /DP	317 2PN /DP	315 2PN /DP	315 2PN/DP mit CP 343-1	416 3PN/DP	412 2PN	412 2PN mit CP 443-1
Telegrammanzahl (Summe)	12	12	12	12	12	12	12
Telegrammanzahl (WinCC)	1	1	1	1	1	1	1
Telegrammanzahl (SPS)	11	11	11	11	11	11	11

Bei „Zykluslesen“ sendet WinCC bei der Testbildanwahl einen Auftrag an die Steuerung, dann antwortet die Steuerung für WinCC zyklisch. WinCC fordert nicht erneut an.

No.	Time	Delta Time display	Source	Destination	Protocol	Length	Info
193	*REF*	*REF*	172.16.70.55	172.16.70.10	T.125	99	detachUserRequest
194	0.062198000	0.062198000	172.16.70.10	172.16.70.55	T.125	295	detachUserRequest
200	2.102248000	2.040050000	172.16.70.10	172.16.70.55	T.125	295	detachUserRequest
204	4.091383000	1.989135000	172.16.70.10	172.16.70.55	T.125	295	detachUserRequest
210	6.080512000	1.989129000	172.16.70.10	172.16.70.55	T.125	295	detachUserRequest
214	8.069638000	1.989126000	172.16.70.10	172.16.70.55	T.125	295	detachUserRequest
218	10.058781000	1.989143000	172.16.70.10	172.16.70.55	T.125	295	detachUserRequest
225	12.098871000	2.040090000	172.16.70.10	172.16.70.55	T.125	295	detachUserRequest
229	14.087975000	1.989104000	172.16.70.10	172.16.70.55	T.125	295	detachUserRequest
241	16.077117000	1.989142000	172.16.70.10	172.16.70.55	T.125	295	detachUserRequest
247	18.066269000	1.989152000	172.16.70.10	172.16.70.55	T.125	295	detachUserRequest
251	20.106362000	2.040093000	172.16.70.10	172.16.70.55	T.125	295	detachUserRequest

Abbildung 5-7: Telegramme zwischen WinCC und S7-317 2PN/DP bei „Zykluslesen bei Änderung“

568	*REF*	*REF*	172.16.70.55	172.16.70.20	T.125	96	detachUserRequest
569	0.001930000	0.001930000	172.16.70.20	172.16.70.55	T.125	296	detachUserRequest

Abbildung 5-8: Telegramme zwischen WinCC und S7-412 2PN bei „Zykluslesen bei Änderung“

Tabelle 5-4: Telegrammanzahl bei „Zykluslesen bei Änderung“

Steuerung / Telegramm	319 3PN /DP	317 2PN /DP	315 2PN /DP	315 2PN/DP mit CP 343-1	416 3PN/DP	412 2PN	412 2PN mit CP 443-1
Telegrammanzahl (Summe)	12	12	12	12	2	2	2
Telegrammanzahl (WinCC)	1	1	1	1	1	1	1
Telegrammanzahl (SPS)	11	11	11	11	1	1	1

1. Bei „Zykluslesen bei Änderung“ verhalten sich die S7-300 und 400 Steuerungen unterschiedlich.
2. S7-300 zeigt das gleiche Verhalten wie bei „Zykluslesen. Die Steuerung sendet zyklisch ein Telegramm an WinCC, unabhängig davon, ob der Wert sich geändert hat oder nicht. „Zur Serien S7-300 bekommt man die gleichen Ergebnisse wie bei „Zykluslesen“.
3. Bei S7-400 Serien sendet WinCC ein Telegramm an die Steuerung. Die Steuerung sendet sofort ein Antworttelegramm mit dem aktuellen Wert zurück. Da der Wert während der Testphase konstant bleibt, erfolgt kein weiterer Telegrammaustausch zwischen WinCC und Steuerung. Also insgesamt gibt es nur 2 Telegramme. Zur Serien S7-400 gibt es nur 2 Telegramme. Am Anfang der Runtime schickt WinCC nach Steuerung einen Auftrag, danach schickt Steuerung eine Antwort zurück. Weil die Messvariable konstant ist, also es nicht sich verändern wird, dann existiert keine Antwort von Steuerung mehr.

<u>Resultat:</u>

Die Messergebnisse zeigen, dass „Zykluslesen bei Änderung“ die höchste Kommunikationseffizienz hat (nur 2 Telegramme). „Zykluslesen“ ist performanter als „Pollen“. Hier wird die Telegramlast um die 50% verringert.“
Nur S7-400 Serien unterstützen „Zykluslesen bei Änderung“. (siehe Tab.5-5)

Tabelle 5-5: Funktionalität vom Lesedienst der gemessenen Steuerungen

Steuerung / Telegramm	319 3PN /DP	317 2PN /DP	315 2PN /DP	315 2PN/DP mit CP 343-1	416 3PN/DP	412 2PN	412 2PN mit CP 443-1
Pollen	x	x	x	x	x	x	x
Zykluslesen	x	x	x	x	x	x	x
Zykluslesen bei Änderung	-	-	-	-	x	x	x

5.2.2 Einstellung der Steuerung

In dieser Messaufgabe wird die Auswirkung von Programmierung und Einstellungen auf der Steuerungsseite auf die Kommunikationsperformance untersucht.

Einflussfaktoren:

- Priorität der BuB-Kommunikation (Bedien- und Beobachtungs-Kommunikation)
- Belastung des Anwenderprogramms
- Einrichtung der Kommunikationslast

Im Abschnitt 4.1.2 wurde den Ablauf des zyklischen Programms erläutert. Kommunikationsbearbeitung wird in jedem Zyklus ausgeführt. Aber wann und wo wird diese ausgeführt? Bei verschiedenen CPUs und Konfigurationen gibt es Differenz, und es hat Einfluss auf die Kommunikationsperformance.
Der Datenzugriff erfolgt mit einem HMI/SCADA (Bedien- und Beobachtungs-Funktionen) bei den S7-300 (bei ohne „priorisierte BuB-Kommunikation“) am Zykluskontrollpunkt. Die Anwenderprogrammbearbeitung wird durch die Bedien- und Beobachtungs-Funktionen nicht unterbrochen. Aber bei der S7-400 werden Kommunikationsaufträge nicht im Zykluskontrollpunkt, sondern in festen Zeitscheiben während des Programmzyklusses bearbeitet (siehe Ab.5-9). Das bedeutet, dass bei S7-400 „BuB Kommunikation“ die höhere Priorität hat.

Hinweis: „priorisierte BuB-Kommunikation“ ist eine neue Funktion für S7-300. Die kann die Priorität der BuB-Kommunikation erhöhen, um die bessere Kommunikation mit HMI/SCADA zu erhalten. Wenn die „priorisierte BuB-Kommunikation“ aktiviert ist, verhält sich die Kommunikationsbearbeitung ähnlich wie bei S7-400, also werden Kommunikationsaufträge in festen Zeitscheiben während des Programmzyklusses bearbeitet. Nicht alle CPUs von S7-300 Serien unterstützen diese Funktion[14]. (die 300 CPUs, die ich für meine Messungen verwendet habe, haben diese Funktionalität unterstützt.)

[14] Die genaue Datei siehe „priorisierte BuB-Kommunikation“ in Siemens Industry Online Surport

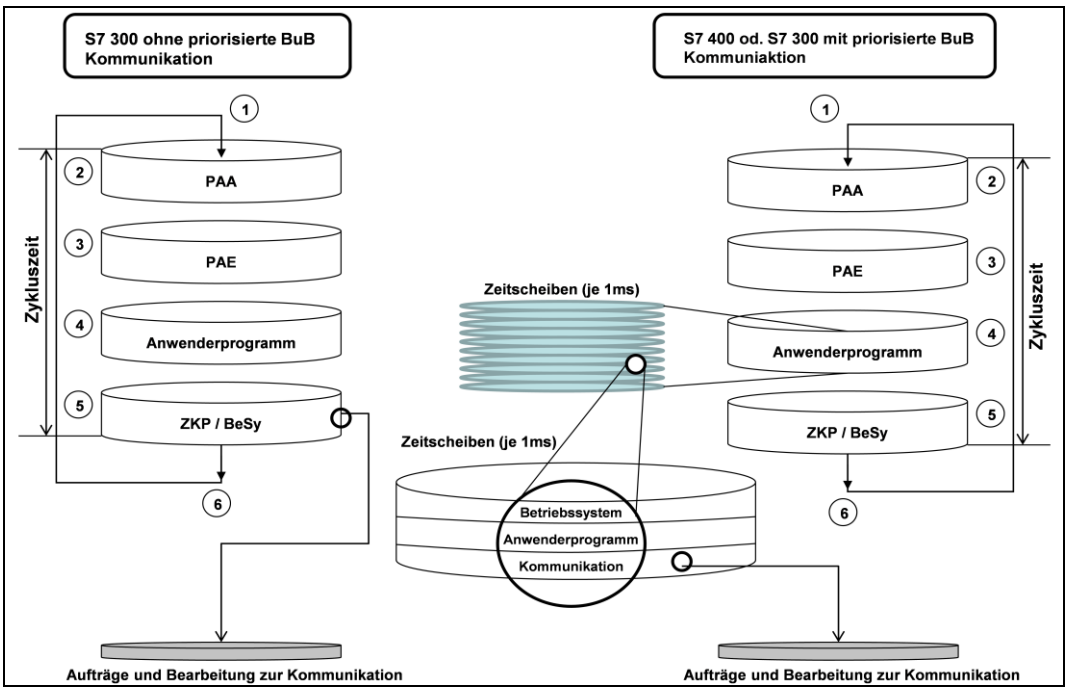


Abbildung 5-9: Bearbeitungsart von Kommunikation bei verschiedenen Status

Wenn die „priorisierte BuB-Kommunikation“ bei S7-300 nicht aktiviert ist, werden die Aufträge und Bearbeitung zur Kommunikation nach Abarbeiten des Anwenderprogramms ausgeführt. Die Bearbeitungszeit des Anwenderprogramms hat einen direkten Einfluss auf die Kommunikation (siehe Ab.5-10). Durch den Funktionblock FB69 (die genaue Dastellung siehe 4.1.2) kann das Anwenderprogramm belastet werden.

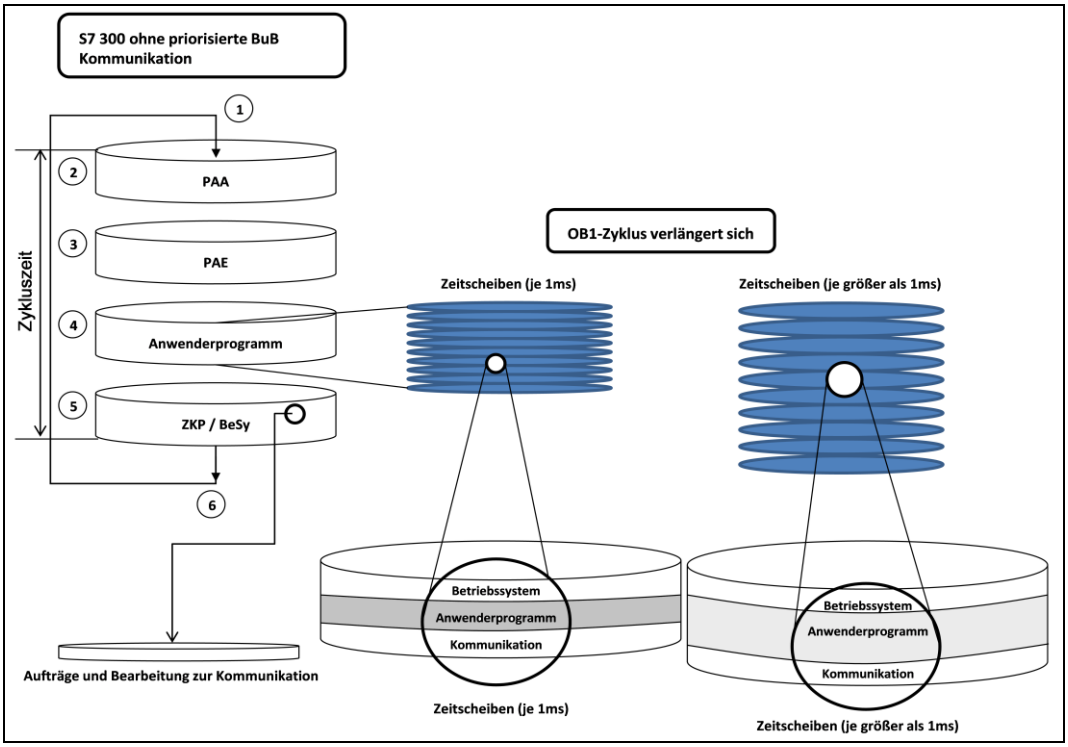


Abbildung 5-10: Verlängerung der Zykluszeit durch Belasten für Anwenderprogramm

Im Dialogfeld: CPU-Eigenschaft gibt es noch ein Parameter „Zyklusbelastung durch Kommunikation“, dieser Parameter hat Auswirkungen auf den Telegramm-Verkehr der CPU. Das Betriebssystem der CPU stellt laufend der Kommunikation den von Ihnen projektierten Prozentsatz der gesamten CPU-Verarbeitungsleistung zur Verfügung (Zeitscheiben-Technik). Also CPU lierfert die Ressource 5% bis 50% für Kommunikationsbelastung.(siehe Ab.5-11) So wird bei diesem Status mehr Ressource von CPU bunutzt, so würde sich die Zykluszeit verlängern.

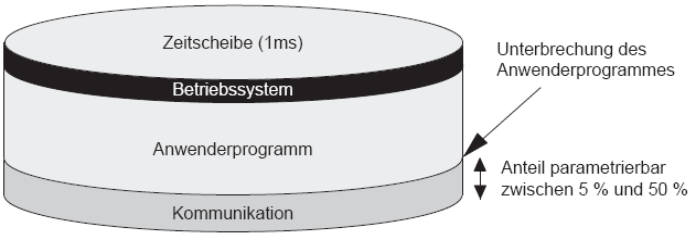


Abbildung 5-11: Aufbau der Zeitscheibe[15]

Messmethode:

Ermitteln der Zugriffszeit von der Anfangsvariable (die Variable, die mit dem ersten Objekt im Bild verbunden ist) bis Endvariable (die Variable, die mit dem letzten Objekt im Bild verbunden ist) im Testbild. Mit Hilfe von Wireshark kann man die Telegramme, die 2 oben erwähnten Variablen enthalten auswählen (filtern). Dadurch das „Delta Time display“ (Zugriffszeit) bekommt wird.
Um die Zugriffszeit einfacher und genauer zu bestimmen, kann das Telegramm inklusiv der Anfangsvariable als Referenztelegramm definiert werden. Dann ist der Zeitstempel des nächsten Telegramms inklusiv der Endevariable die Zugriffszeit. (siehe Ab.5-12)

6134	*REF*	*REF*	172.16.70.55	172.16.70.10	T.125	85	detachUserRequest
7841	15.466883000	15.466883000	172.16.70.55	172.16.70.10	T.125	85	detachUserRequest
7844	*REF*	*REF*	172.16.70.55	172.16.70.10	T.125	85	detachUserRequest
9537	15.466785000	15.466785000	172.16.70.55	172.16.70.10	T.125	85	detachUserRequest
9542	*REF*	*REF*	172.16.70.55	172.16.70.10	T.125	85	detachUserRequest
11242	15.476861000	15.476861000	172.16.70.55	172.16.70.10	T.125	85	detachUserRequest

Abbildung 5-12: Telegramm zur S7-317 2PN/DP bei OB1 Zyklus: 30ms

Konfiguration:

Im Dialogfeld CPU-Eigenschaft können „priorisierte BuB-Kommunikation“ (nur bei S7-300 Serien) und „Zyklusbelastung durch Kommunikation“ eingestellt werden.

[15] Quelle: Automatisierungssystem S7-400 CPU-Daten Gerätehandbuch: 9.4 Seite 214

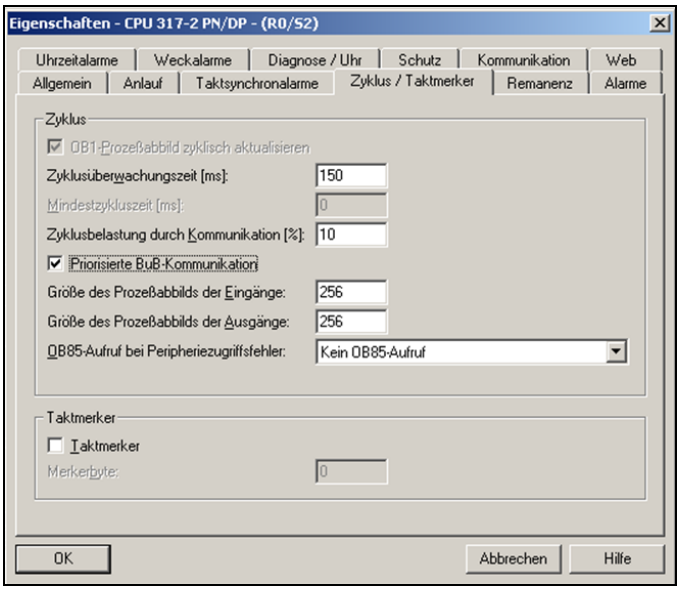


Abbildung 5-13: Dialogfeld: Eigenschaften von CPU

In FC6 kann die Zykluszeit eingeben werden. Die Zeit darf nicht über 130ms (in unserem Test) eingestellt werden, sonst geht die CPU in stop. (Der gegebene Parameter ist nur die Anwenderprogrammbearbeitungszeit. Die Zeit des ganzen Zykluses (Anwenderprogramm, Betriebssystem und Kommunikation) ist ein weinig größer als 130ms. Ist der Parameter über 130ms eingestellt, wird der ganze Zyklus über max. Überwachungszeit (150ms) liegen.

```
// Einstellung der OB1 Zykluszeit

      CALL  FB    69 , DB69
       LoadTime:=T#30MS                    // im Bereich von 10ms bis 130ms
```

Abbildung 5-14: Einstellung vom OB1 Zyklus in FC6

Die nachfolgende Tabelle zeigt die Einstellungen von WinCC und Wireshark bei dieser Messaufgabe.

Tabelle 5-6: Einstellung von WinCC und Wireshark zum Test „Einstellung der Steuerung“

WinCC	Verbindungsart	Lesedienstart	Testbild	Aktualisierungszyklus der Objekt
	TCP/IP	Pollen	TB2, TB3	100ms
Wireshark	Anweisung zum Greifen der Telegramme			
	frame[-5:5]==1f:84:00:00:00||frame[-5:5]==20:84:06:23:d0 bei S7-300 TB2 frame[-5:5]==ca:00:1f:00:ca||frame[-5:5]==ca:00:20:c4:7a bei S7-400 TB2 frame[-5:5]==ca:00:1f:00:ca||frame[-5:5]==ca:00:22:c4:7a bei S7-400 TB3			

5.2.2.1 Einstellung des OB1 Zyklus und BuB-Kommunikation

Messumgebung:

Tabelle 5-7: Einstellung der Parameter „OB Zyklus“, „Priorisierte BuB-Kommunikation“

Steuerung	**OB1 Zyklus**	**Priorisierte BuB-Kommunikation**	Zyklusbelastung durch Kommunikation
	30ms, 50ms, 130ms	**deaktiv / aktiv**	20%

Messergebnis:

priorisierte BuB Kommunikation: deaktiv

Tabelle 5-8: Zugriffszeit bei OB1 Zyklus: 30ms (deaktiv)

Zugriffszeit	319 3PN/DP	317 2PN/DP	315 2PN/DP	315 2PN/DP mit CP 343-1
Einheit: [s]	500 Variablen von DB31Var0 bis DB32Var249			
Manual	15.5	15.9	15.4	23.0
durch Wireshark	15.5	15.5	15.5	23.0

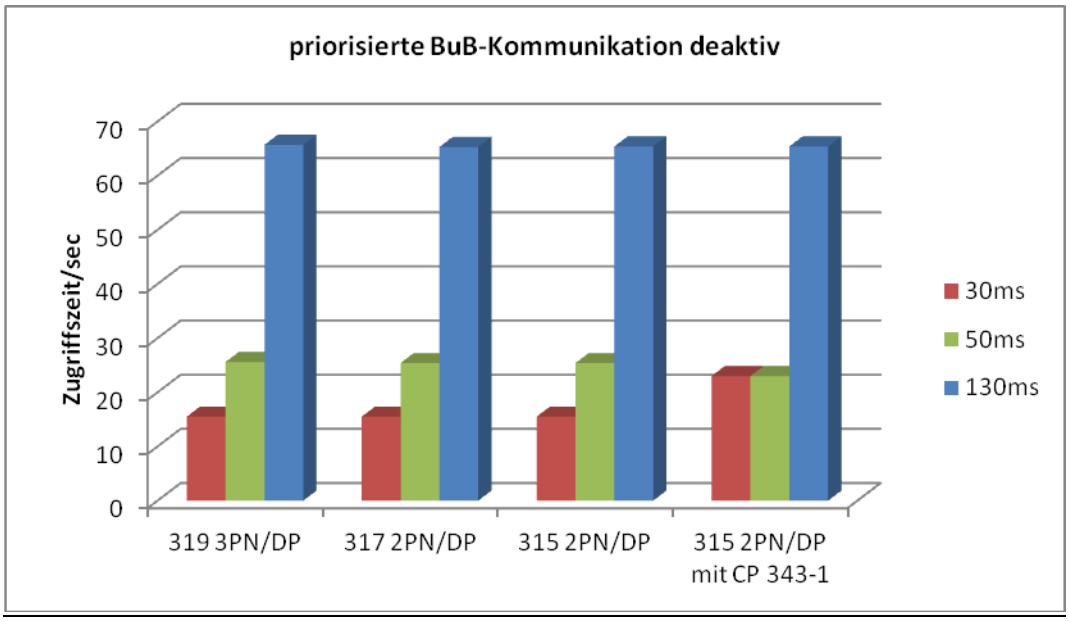
Tabelle 5-9: Zugriffszeit bei OB1 Zyklus: 50ms (deaktiv)

Zugriffszeit	319 3PN/DP	317 2PN/DP	315 2PN/DP	315 2PN/DP mit CP 343-1
Einheit: [s]	500 Variablen von DB31Var0 bis DB32Var249			
Manual	25.9	25.9	25.8	23.0
durch Wireshark	25.6	25.4	25.4	23.0

Tabelle 5-10: Zugriffszeit bei OB1 Zyklus: 130ms (deaktiv)

Zugriffszeit	319 3PN/DP	317 2PN/DP	315 2PN/DP	315 2PN/DP mit CP 343-1
Einheit: [s]	500 Variablen von DB31Var0 bis DB32Var249			
Manual	66.2	65.7	65.8	66.5
durch Wireshark	65.7	65.3	65.4	65.4

Diagramm 5-1: Zugriffszeit verschiedener CPUs und CPs bei OB1 Zyklus: 30, 50, 130ms (ohne „priorisierte BuB-Kommunikation“)



priorisierte BuB Kommunikation: aktiv (gültig für S7-300, aber bei S7-400 ähnlich)

Tabelle 5-11: Zugriffszeit bei OB1 Zyklus: 30ms (aktiv)

Steuerung / Zugriffszeit	319 3PN /DP	317 2PN /DP	315 2PN /DP	315 2PN/DP mit CP 343-1	416 3PN/DP	412 2PN	412 2PN mit CP 443-1
Einheit: [s]	500 Variablen von DB31Var0 bis DB32Var249						
Manual	5.0	5.0	5.0	15.7	2.7	2.4	2.5
durch Wireshark	5.0	5.0	5.0	15.7	2.5	2.5	2.5
Einheit [s]	1000 Variablen von DB31Var0 bis DB34Var249						
Manual	-	-	-	-	5.0	4.9	5.0
durch Wireshark	-	-	-	-	5.0	5.0	5.0

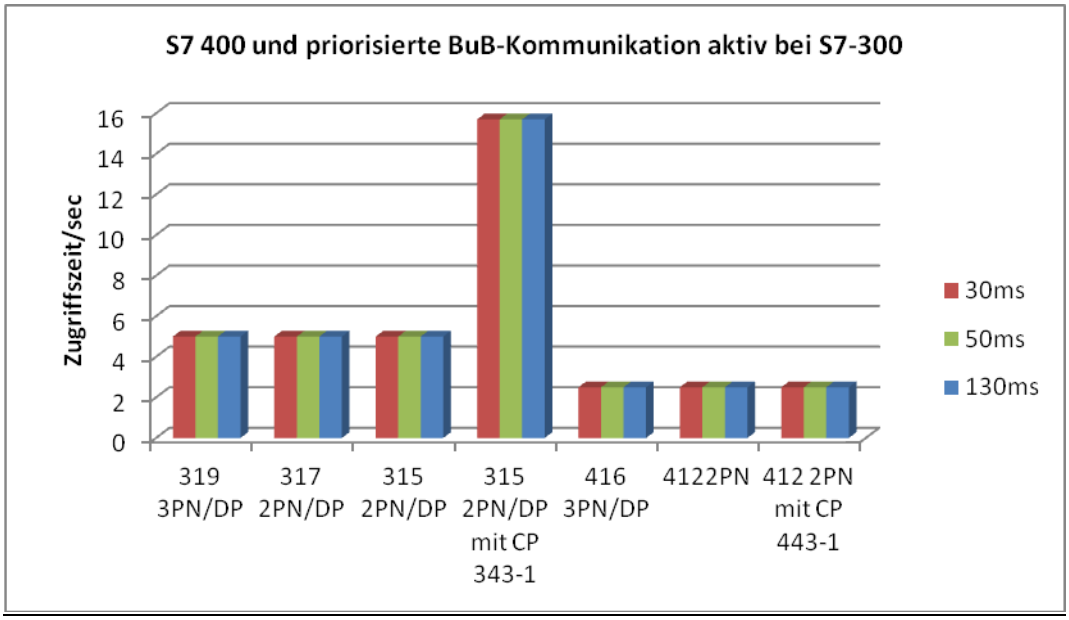
Tabelle 5-12: Zugriffszeit bei OB1 Zyklus: 50ms (aktiv)

Steuerung / Zugriffszeit	319 3PN /DP	317 2PN /DP	315 2PN /DP	315 2PN/DP mit CP 343-1	416 3PN/DP	412 2PN	412 2PN mit CP 443-1
Einheit: [s]	500 Variablen von DB31Var0 bis DB32Var249						
Manual	5.0	5.0	5.0	15.7	2.7	2.4	2.5
durch Wireshark	5.0	5.0	5.0	15.7	2.5	2.5	2.5
Einheit [s]	1000 Variablen von DB31Var0 bis DB34Var249						
Manual	-	-	-	-	5.0	4.9	5.0
durch Wireshark	-	-	-	-	5.0	5.0	5.0

Tabelle 5-13: Zugriffszeit bei OB1 Zyklus: 130ms (aktiv)

Steuerung / Zugriffszeit	319 3PN /DP	317 2PN /DP	315 2PN /DP	315 2PN/DP mit CP 343-1	416 3PN/DP	412 2PN	412 2PN mit CP 443-1
Einheit: [s]	500 Variablen von DB31Var0 bis DB32Var249						
Manual	5.0	5.0	5.0	15.7	2.7	2.4	2.5
durch Wireshark	5.0	5.0	5.0	15.7	2.5	2.5	2.5
Einheit [s]	1000 Variablen von DB31Var0 bis DB34Var249						
Manual	-	-	-	-	5.0	4.9	5.0
durch Wireshark	-	-	-	-	5.0	5.0	5.0

Diagramm 5-2: Zugriffszeit verschiedener CPUs und CPs bei OB1 Zyklus: 30, 50, 130ms (mit „priorisierte BuB-Kommunikation“)



Wird die „priorisierte BuB Kommunikation“ bei S7-300 nicht aktiviert, hat die Zykluszeit von OB1 den großen Einfluss auf die Zugriffszeit. Die Zugriffszeit erhöht sich mit zunehmendem Zyklus von OB1.

Wenn man „priorisierte BuB Kommunikation“ aktiviert ist, verringert sich die Zugriffszeit erheblich. Dabei hat der OB1 Zyklus keinen Einfluss auf die Zugriffszeit. S7-400 arbeitet intern mit „priorisierte BuB Kommunikation“ Funktionalität.

Auch trotz der aktivierten „priorisierte BuB Kommunikation“ bei S7-300 ist die Zugriffszeit bei S7-400 um die Hälfte kleiner als bei S7-300. Diese wird durch die Größe der PDU-Size bei S7-400 (PDU-Size = 408 Bytes) verursacht. Bei S7-300 ist die PDU-Size 204 Bytes.

5.2.2.2 Einstellung von Zyklusbelastung durch Kommunikation:

Messumgebung:

Tabelle 5-14: Erstellung des Parameters: „Zyklusbelastung durch Kommunikation“

Steuerung	OB1 Zyklus	Priorisierte BuB-Kommunikation	**Zyklusbelastung durch Kommunikation**
	50ms	aktiv	**10%, 20%, 50%**

Tabelle 5-15: Zugriffszeit bei Zyklusbelastung durch Kommunikation 10%

Steuerung / **Zugriffszeit**	319 3PN /DP	317 2PN /DP	315 2PN /DP	315 2PN/DP mit CP 343-1	416 3PN/DP	412 2PN	412 2PN mit CP 443-1
Einheit: [s]	500 Variablen von DB31Var0 bis DB32Var249						
Manual	5.0	5.2	5.1	15.4	2.7	2.5	2.7
durch Wireshark	5.0	5.0	5.0	15.7	2.5	2.5	2.5
Einheit: [s]	1000 Variablen von DB31Var0 bis DB34Var249						
Manual	-	-	-	-	5.1	5.2	5.1
durch Wireshark	-	-	-	-	5.0	5.0	5.0

Tabelle 5-16: Zugriffszeit bei Zyklusbelastung durch Kommunikation 20%

Steuerung / **Zugriffszeit**	319 3PN /DP	317 2PN /DP	315 2PN /DP	315 2PN/DP mit CP 343-1	416 3PN/DP	412 2PN	412 2PN mit CP 443-1
Einheit: [s]	500 Variablen von DB31Var0 bis DB32Var249						
Manual	5.2	5.3	5.2	15.6	2.6	2.6	2.6
durch Wireshark	5.0	5.0	5.0	15.7	2.5	2.5	2.5
Einheit: [s]	1000 Variablen von DB31Var0 bis DB34Var249						
Manual	-	-	-	-	5.0	5.0	5.4
durch Wireshark	-	-	-	-	5.0	5.0	5.0

Tabelle 5-17: Zugriffszeit bei Zyklusbelastung durch Kommunikation 50%

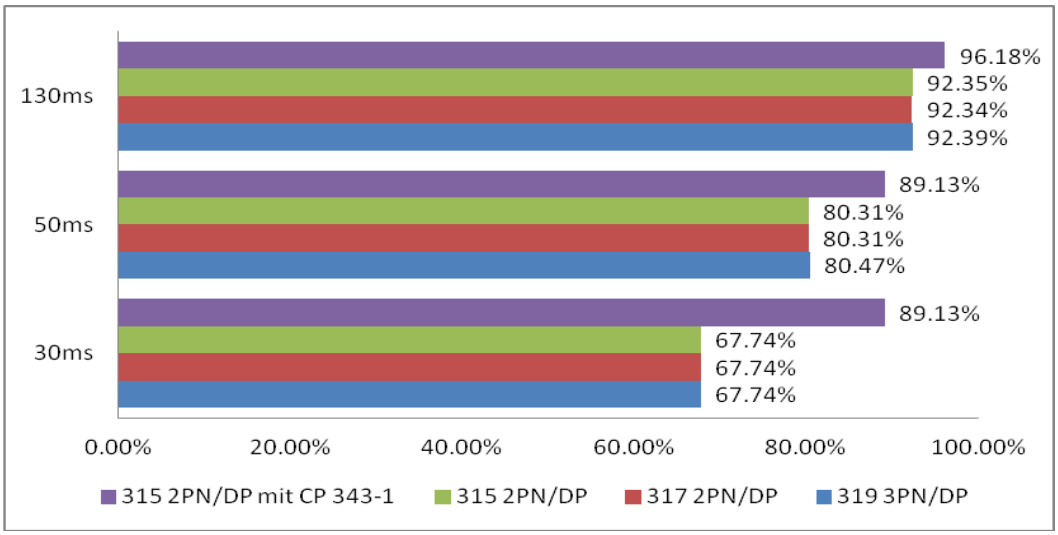
Steuerung / **Zugriffszeit**	319 3PN /DP	317 2PN /DP	315 2PN /DP	315 2PN/DP mit CP 343-1	416 3PN/DP	412 2PN	412 2PN mit CP 443-1
Einheit: [s]	500 Variablen von DB31Var0 bis DB32Var249						
Manual	5.2	5.2	5.2	15.3	2.6	2.8	2.6
durch Wireshark	5.0	5.0	5.0	15.3	2.5	2.5	2.5
Einheit: [s]	1000 Variablen von DB31Var0 bis DB34Var249						
Manual	-	-	-	-	5.0	5.1	5.1
durch Wireshark	-	-	-	-	5.0	5.0	5.0

Der Parameter „Zyklusbelastung durch Kommunikation“ hat kaum Einfluss auf die Kommunikation im Test. Die Änderungen der „Zyklusbelastung durch Kommunikation“ hat in unserem Test keine Performance-Verbesserung bzw. -Verschlechterung bewirkt.

Resultat:

1. Die Funktion: „priorisierte BuB Kommunikation“ hat den großten Einfluss auf die Kommunikation zwischen SPS und WinCC. Wenn diese bei S7- 300 aktiviert wird, erhöht sich die Kommunikationsperformance erheblich. Diag.5-3 zeigt die Prozentzahl der Erhöhung von der Kommunikation beim verschiedenen OB1 Zyklus, wenn „priorisierte BuB Kommunikation“ aktiviert. z.B. CPU 315 2PN/DP bei OB1 Zyklus: 130ms verringert sich die Zugriffszeit um 92.35%, also die Kommunikationsperformance erhöht sich um 92.35%.
2. Die Zykluszeit von OB1 hat ein Einfluss auf S7-300, wenn „priorisierte BuB Kommunikation“ nicht aktiviert ist. (Je länger der OB1 Zyklus, desto schlechter die Kommunikationsperformance) Die „priorisierte BuB Kommunikation“ hat keinen Einfluss auf die Kommunikationsperformance.
3. Trotz der Funktionalität „priorisierte BuB Kommunikation“ bei S7-300 CPUs, ist die Kommunikationperformance bei S7-400 CPUs viel höher. Dies berührt darauf, dass S7-400 CPUS 2 Prozessoren und S7-300 CPUs nur einen Prozessor haben. Durch das Vorhandensein von zwei Prozessoren bei S7-400 CPUs können die Resourcen optimal auf beide Prozessoren aufgeteilt werden.
4. Der Parameter „Zyklusbelastung durch Kommunikation“ hat keine Auswirkung auf die Optimierung der Kommunikation im Test. Die möglichen Ursachen sind: Die reale Kommunikationslast von der CPU war geringer als die projektierte Kommunikationslast. Deshalb konnten beim Test die Kommunikationsressourcen der CPU nicht voll ausgenutzt werden. Aus diesem Grund konnten nicht die Auswirkungen von „Zyklusbelastung durch Kommunikation“ auf die Performance festgestellt werden.

Diagramm 5-3: Erhöhung der Kommunikationsperformance mit „priorisierte BuB Kommunikation“



5.2.3 Auswirkung des Bildaufbaus und Konfiguration von WinCC

In dieser Messaufgabe wird die WinCC Kommunikationsperformance, aufgrund von Bildaufbau und Konfiguration von WinCC untersucht.

Einflussfaktoren:

- Dynamisierungsart
- Triggerart
- Gemischte Trigger
- Objektanzahl
- Belastung von Logging

Messmethode.

In den weiteren Messaufgaben wird Wireshark benutzt, um die gesendeten und empfangenen Telegramme von WinCC zu filtern, dann fassen wir die Daten (z.B. Zeit; Aktualisierungszyklus) der Telegramme zusammen. Mit Hilfe des Diagramms analysiert man den Ablauf und die Kommunikationsperformance von WinCC, zuletzt bekommt man eine genaue Darstellung der Ergebnisse.

Im Abschnitt 2.1.2 wurde vorgestellt, dass die Kommunikation zwischen WinCC und SPS durch Austauch von Telegrammen erfolgt. Hier wird der Aufbau des gesendeten Telegramms von WinCC dargestellt. Die nachfolgende Abbildung zeigt ein Telegramm mit Protokoll TCP/IP, das die Informationen der Servernamen, MAC-Aderesse, IP-Aderesse, Länger der Daten usw. erhält. Für die Messaufgabe ist die Information von Aplication Layer[16] (Ab. 5-15) relevant.

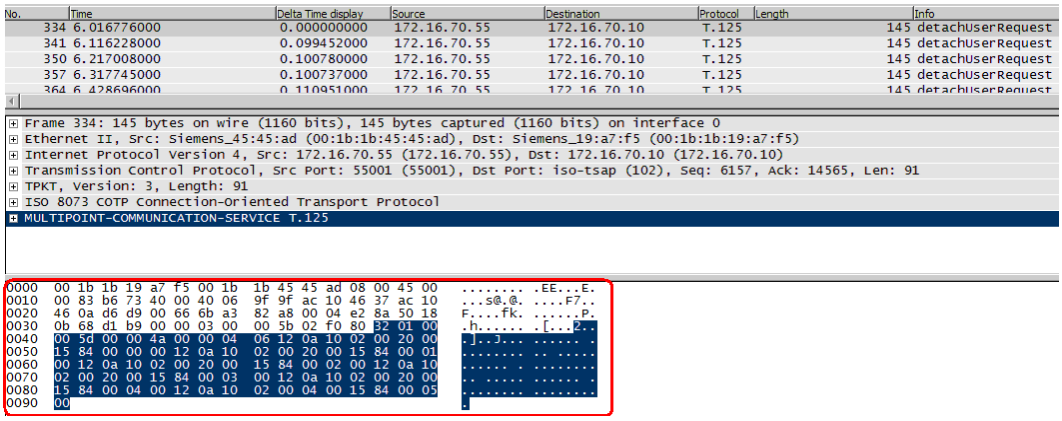


Abbildung 5-15: Auftrag von WinCC nach S7-317 2PN/DP

Die Ziffern mit dunkel-blauer Farbe in der obigen Abbildung sind die Daten der Leseaufträge der Prozessvariablen. Die nützlichen Daten sind die Ziffern von Zelle „0040“ bis „0090“

0040 xx xx xx xx xx xx xx xx xx xx xx 10 02 00 20 00 .¥..J... ..... .

0050 15 84 00 00 00 12 0a 10 02 00 20 00 15 84 00 01 ....

0060 00 12 0a 10 02 00 20 00 15 84 00 02 00 12 0a 10 ....

0070 02 00 20 00 15 84 00 03 00 12 0a 10 02 00 20 00 .. ..

[16] Siehe ISO-Reference-Model

0080 15 84 00 04 00 12 0a 10 02 00 04 00 15 84 00 05 ....

0090 00

10: SyntaxID, bei Simatic immer 10

02: Datentyp, das bedeutet in diesem Fall werden Byte angeben

00 20: Länger der Daten angegeben, in diesem Fall sind das Hexadezimal 20 was 32 Dezimal entspricht, also 32 Byte.

0015: DB-Nummer, in diesem Fall sind das Hexadezimal 15 was 21, also DB21.

84 00 00 00: die Byte und Bit Aderesse der Variable (z.B. aktuelle Variable: Byte0, Bit0)

Zum anderen Beispiel: 84 00 02 00 HEX -> Binär: 0000 0000, 0000 0010, 0000 0000 nach rechts 3 Ziffern verschieben (Die letzte 3 Ziffern bedeutet, was die Bitadresse ist.)

0000 0000, 0000 0000, 0100 0000 Binär -> Dezimal: 64, also Byte Nummer 64

Hinweis: Für weitere Information siehe „Any-Pointer“ in die Hilfe zu STEP7

Wenn die letzte 10 Ziffer in der Filter als die Form z.B. frame[-5:5]==15:84:00:05:00 eingeben wird, kann das Telegramm durch Wireshark ausgewählt werden.

Welches Telegramm soll ausgewählt wird?

Ein Auftrag (Telegramm) von WinCC kann max. 46 DWORD Variablen bei S7-300, 108 bei S7-400 verpacken. Die genaue Information (Adresse, Länge usw.) der Leseaufträge mit den Prozessvariablen kann vom Telegramm bekommen wird. So soll das ausgewählte Telegramm nachfolgende Eigenschaften erhalten:

1. Es erhält die Prozessvariablen, die mit der Testobjekte im Bild verbunden werden.
2. Es ist vollständig. (bei S7-300: Die Leseaufträge erhält 46 DWORD Prozessvariablen pro Telegramm. bei S7-400: 108 DWORD Prozessvariablen pro Telegramm)

Die Parameter „Delta Time display“ und „Time“ (siehe Ab. 5-2) sind die Messparameter. Der Grundsatz dieser Methode ist: Die Leseanforderungen mit Prozessvariablen werden zyklisch von WinCC gesendet. (Zykluszeit ist abhängig von der vorgegebenen Aktualisierungszeit.) Durch Filter werden die gleiche Telegramme (Aufträge) ausgewählt, und im Testfenster von Wireshark angezeigt. „Delta Time display“ ist der tatsächliche Aktualisierungszyklus. Die ausgewählten Telegrammen kann als CVS-Datei gespeichert werden. Nach der Bearbeitung der Datei kann ein Diagramm mit den Parametern „Zeit“ und „Aktualisierungszyklus“ angefertigt werden. Aus dem Diagramm kann die genaue Aktualisierungszeit im Messintervall abgelesen werden. Aus dem Diagremm kann die Kommunikationsstaus analysiert werden z. B, ob es ein Kommunikationslast in diesem Messintervall aufgetreten war (tatsächlicher aktualisierungszyklus ist größer als vorgegebene Zykluszeit).

Das Verhältnis eines Telegramms ist repräsentativ für allen anderen Telegrammen. D.h. Die Ergebnisse und Analyse ein einziges Telegramm kann auf die ganzen Telegrammen im Messintervall bezüglich der Kommunikationsperformance angewendet werden.

Hinweis: Bei Pollen senden WinCC und Steuerungen die Telegramme miteinander, so existiert es im Diagnosefenster von Wireshark immer gleichzeitig ein Telegrammpaar. Da der Verlauf des Telegramms vom WinCC und Steuerung fast identisch, werden im Wireshark wegen der Einfachheit und Übersichtichkeit nur Telegramme von WinCC (Leseaufträge) ausgewählt und analysiert.

Messumgebung bei Steuerung:

Für die weitere Messaufgabe werden die Parameter von Steuerung wie die nachfolgende Tabelle eingestellt.

Tabelle 5-18: Allgemeine Konfiguration der Steuerung

Steuerung	OB1 Zyklus	Priorisierte BuB-Kommunikation	Zyklusbelastung durch Kommunikation
	50ms	deaktiv	20%

5.2.3.1 Untersuchung der Dynamisierungsarten

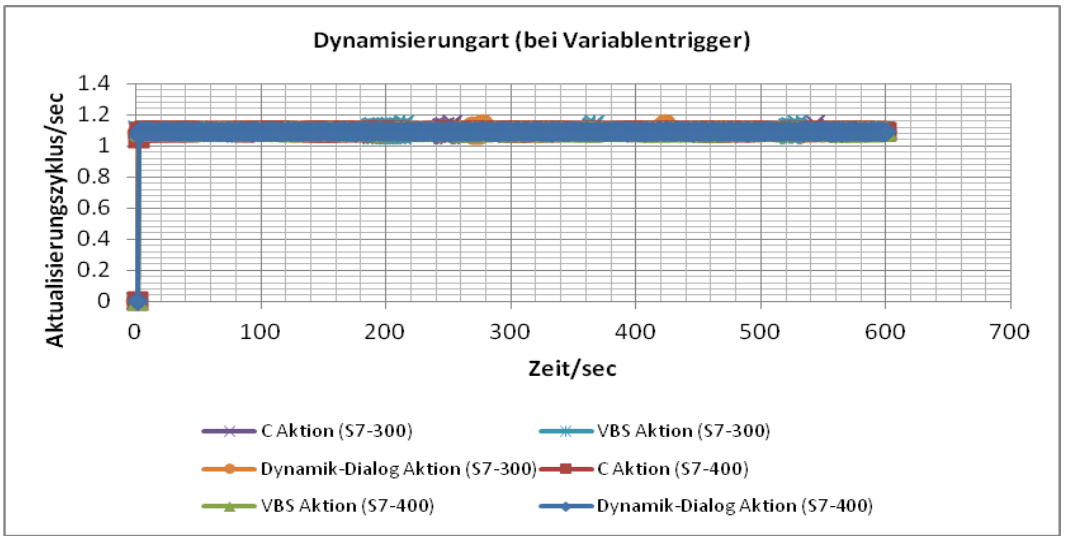
Messumgebung:

Tabelle 5-19: Einstellung von WinCC und Wireshark zum Test: „Dynamisierungsart“

WinCC	Verbindungsart	Lesedienstart	**Testbild**	Aktualisierungszyklus	Logging
	TCP/IP	pollen	**TB4, TB5, TB6; TB7, TB8, TB9**	1000ms	deaktiv
Wireshark	Messintervall	Anweisung zum Greifen der Telegramme[17]			
	600sec	**[1]frame[-5:5]==15:84:00:05:00** bei S7-300 für alle Testbilder **[4]frame[-5:5]==20:00:16:00:00** bei S7-400 für alle Testbilder			

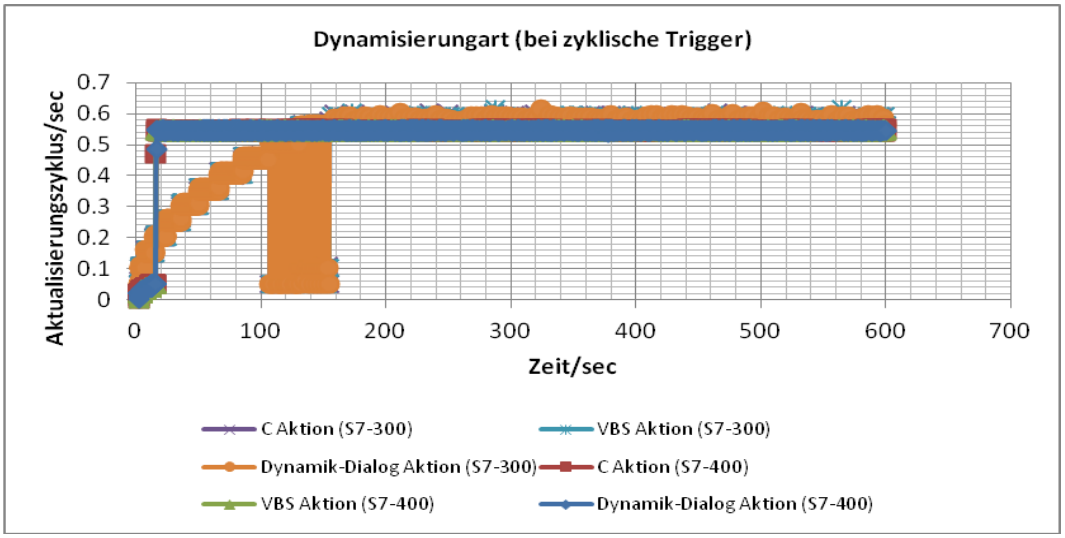
Messergebnis:

Diagramm 5-4: Verschiedene Dynamisierungsarten bei Variablentrigger



[17] Die ausführliche Darstellung der Anweisungen siehe A[03] im Anhang

Diagramm 5-5: Verschiedene Dynamisierungsarten bei zyklischen Trigger



Die verschiedenen Arten von Dynamisierungen (C, VBS, …) in Bildern bei gleichen Triggerarten (z.B. Zyklischer Trigger) zeigen den gleichen Verlauf.
Also die Art von Dynamisierung hatte in unserem Test keinen Einfluss auf die Kommunikationsperformance von WinCC.

5.2.3.2 Untersuchung der Triggerarten

<u>Messumgebung:</u>

Tabelle 5-20: Einstellung von WinCC und Wireshark zum Test: „Triggerart“

WinCC	Verbindungsart	Lesedienstart	**Testbild**	Aktualisierungszyklus	Logging
	TCP/IP	Pollen	**TB4, TB7**	1000ms	deaktiv
Wireshark	Messintervall	Anweisung zum Greifen der Telegramme			
	600sec	**[1]frame[-5:5]==15:84:00:05:00** bei S7-300 für TB4, TB7 **[4]frame[-5:5]==20:00:16:00:00** bei S7-400 für TB4, TB7			

Messergebnis:

Diagramm 5-6: Zyklischer Trigger, Dynamisierung mit C-Aktion (S7-300)

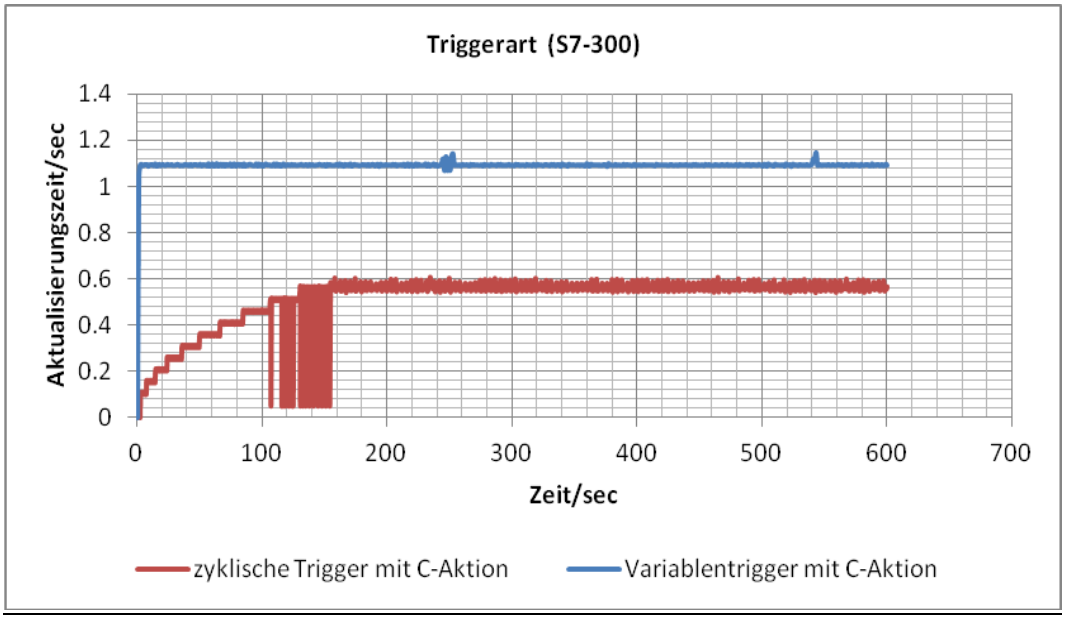
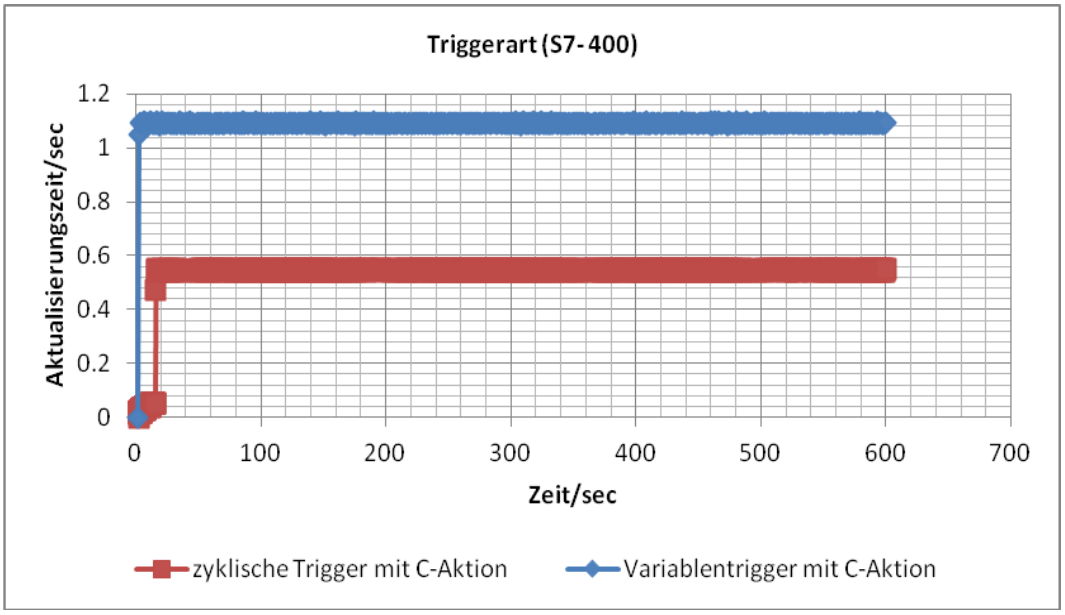


Diagramm 5-7: Variablentrigger, Dynamisierung mit C-Aktion (S7-400)



Aus den obigen 2 Diagrammen kann man sehen:

Die Tendenz der Linien im Diagramm von zyklischen Trigger und Variablentrigger haben die große Differenz. Nach Beobachten und Analyse von den Telegrammen wird dafür eine Erklärung gegeben. (Um die nachfolgende Inhalte einfach zu verstehen, kann das Flussdiagramm: A [05] im Anhang gelesen werden.)

Wenn in einem Bild Aktionen mit zyklischem Trigger projektiert sind, wird bei Bildanwahl jeder Variable von WinCC Datenmanager einzeln angefordert. D.h. für jede angeforderte Vairable sendet der WinCC Datenmanager ein Telegramm an die Steuerung. Das erste

Telegramm beinhaltet die Variable 1. Das zweite Telegramm beinhaltet die Variable1 und 2. Und so weiter. Bis die maximale Telegrammlänge (46 DWORD Variablen) erreicht ist. Danach werden die angeforderten Variablen in 2 Telegramme verpackt und an die Steuerung gesendet bis wiederum die maximale Länge von Telegramm 2 erreicht ist. Dann werden die Variablen in 3 Telegramme verpackt usw.
Die Zykluszeiten (Delta Time display) werden mittels Wireshark erfasst und ausgewertet. Am Anfang sind die angeforderten Telegramme niedrig. Deshalb ist die Zykluszeit klein. Mit der Zunahme von Leseaufträge wächst dementsprechend auch die Zykluszeit (siehe Diag.5-6, 5-7). Die Zykluszeit bleibt dann konstant, nachdem alle Variablen am Datenmanager angemeldet sind.
Während der sogenannten „Reaktionszeit“ (die Zeit bis die Kurve ihre Sättigung erreicht hat) werden die Dynamisierungen im Bild (z.B. Hintergrundfarbe der Objekte) nicht aktualisiert. Die „Reaktionszeit“ betrug bei S7-300 Serien ca. 156 sec und bei S7-400 Serien ca. 15 sec. Danach beträgt der Aktualisierungszyklus die Hälfte des vorgegebenen Wertes (in diesem Fall 500ms).
Wenn im Bild Aktionen mit Variablentirgger projektiert sind, werden beim Bildanwahl alle Triggervariablen einer Aktion blockweise angefordert. Also es gibt keine Reaktionszeit beim Bildanwahl. Der Aktualisierungszyklus ist fast gleich wie die vorgegebene Wert.

5.2.3.3 Gemischte Trigger

Messumgebung:

Tabelle 5-21: Einstellung von WinCC und Wireshark zum Test: „Gemischte Trigger“

WinCC	Verbindungsart	Lesedienstart	**Testbild**	Aktualisierungszyklus	Logging
	TCP/IP	Pollen	**TB10, TB13**	1000ms	deaktiv
Wireshark	Messintervall	Anweisung zum Greifen der Telegramme			
	600sec	**[2]frame[-5:5]==1a:84:00:05:00** bei S7-300 TB10 **[5]frame[-5:5]==20:00:1b:00:00** bei S7-400 TB10 **[3]frame[-5:5]==1a:84:00:05:c0** bei S7-300 TB13 **[6]frame[-5:5]==20:00:1b:00:a0** bei S7-300 TB13			

Messergebnis:

Diagramm 5-8: EA_Feld im Bild bei gemischten Trigger bei S7-300

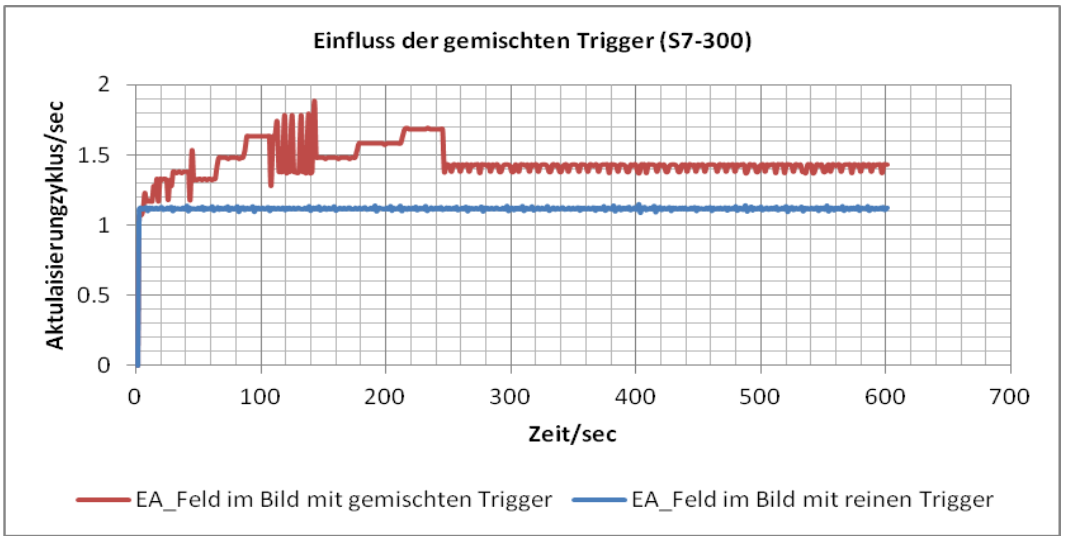
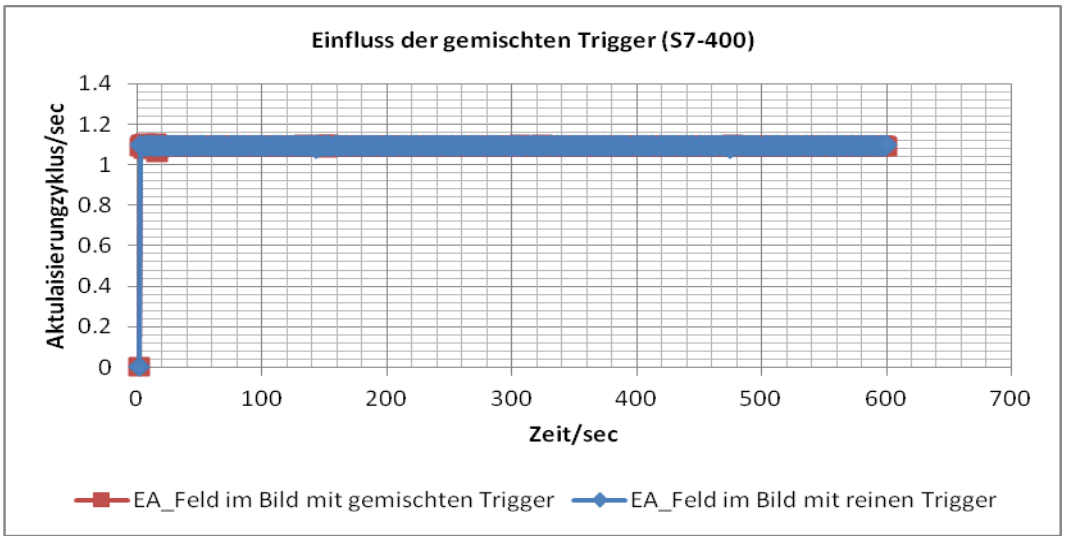


Diagramm 5-9: EA_Feld im Bild bei gemischten Trigger bei S7-400



Beim gemischten Trigger (EA_Felder mit Variablentrigger und Rechtecke mit zyklischen Trigger) zeigt sich der Aktualisierungszyklus von Testobjekt am Anfang einen unstabielen Verlauf im Zeitbereich von 0s bis 245s (Diag.5-8). Der Zeitbreich bis zur „Sättigung“ ist fast gleich wie der Objekte mit dem reinen zyklischen Trigger. Nach diesem Zeitbereich ist die wirkliche Aktualisierungszeit von Testobjekt (1400ms) größer als der vorgegebne Wert (1000ms).

Wenn alle Dynamisierungen im Testbild mit Variablentrigger projektiert werden, ist die Aktualisierungszykluszeit stabiel und fast gleich des vorgegebenen Wertes.

Bei S7-400 gibt es fasst keinen Einfluss, ob die Objekte im Bild mit gemischten Trigger projektiert werden (Diag.5-9).

5.2.3.4 Anzahl der Objekte (Prozessvariablen) im Bild

Messumgebung:

Tabelle 5-22: Einstellung von WinCC, Wireshark zum Test: „Anzahl der Objekte im Bild“

WinCC	Verbindungsart	Lesedienstart	**Testbild**	Aktualisierungszyklus	Logging
	TCP/IP	Pollen	**TB10, TB16**	1000ms	deaktiv
Wireshark	Messintervall	Anweisung zum Greifen der Telegramme			
	600sec	**[1]frame[-5:5]==15:84:00:05:00**bei S7-300 TB10, TB16 **[4]frame[-5:5]==20:00:16:00:00**bei S7-400 TB10, TB16			

Messergebnis:

Diagramm 5-10: Rechteck im Bild mit 250 und 500 Objekte (S7-300 bei Variablentrigger)

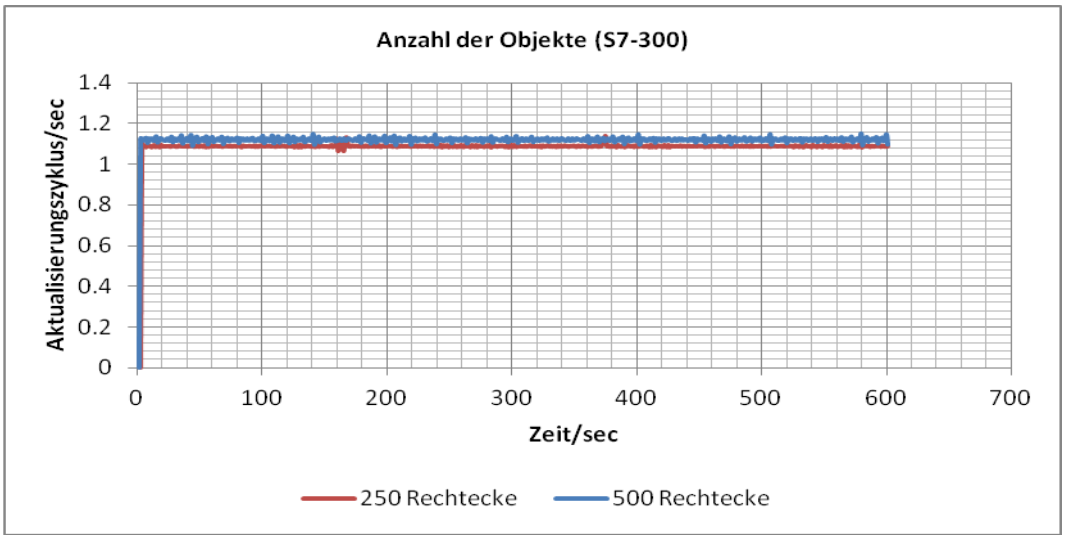


Diagramm 5-11: Rechteck im Bild mit 250 und 500 Objekte (S7-400 bei Variablentrigger)

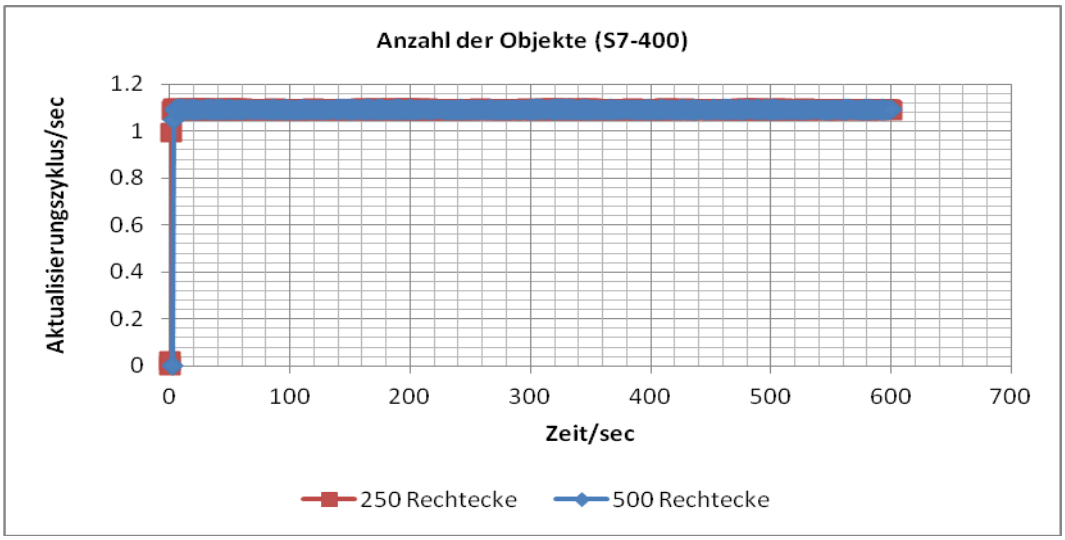


Diagramm 5-12: Rechteck im Bild mit 250 und 500 Objekte (S7-300 bei zyklischen Trigger)

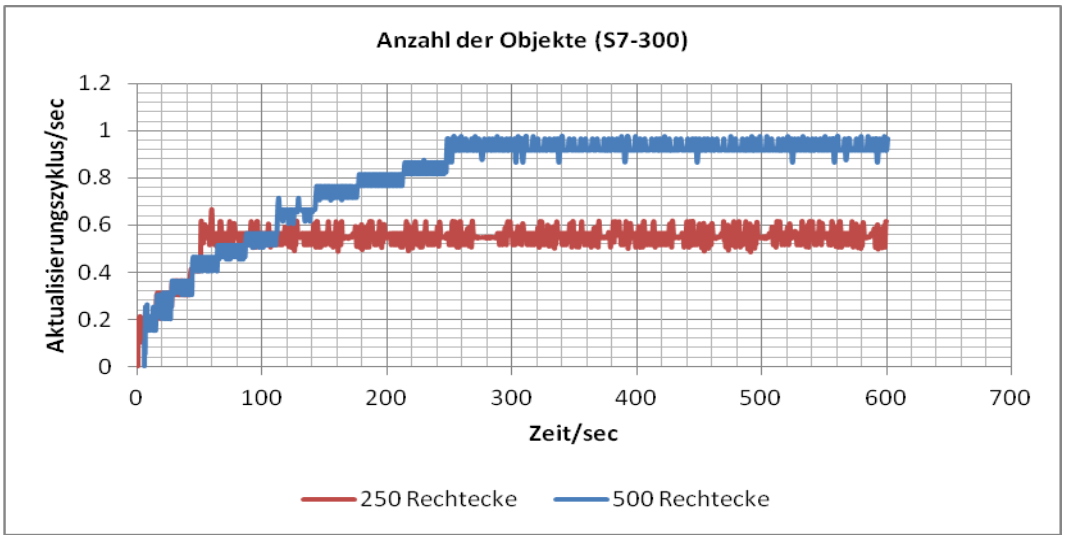
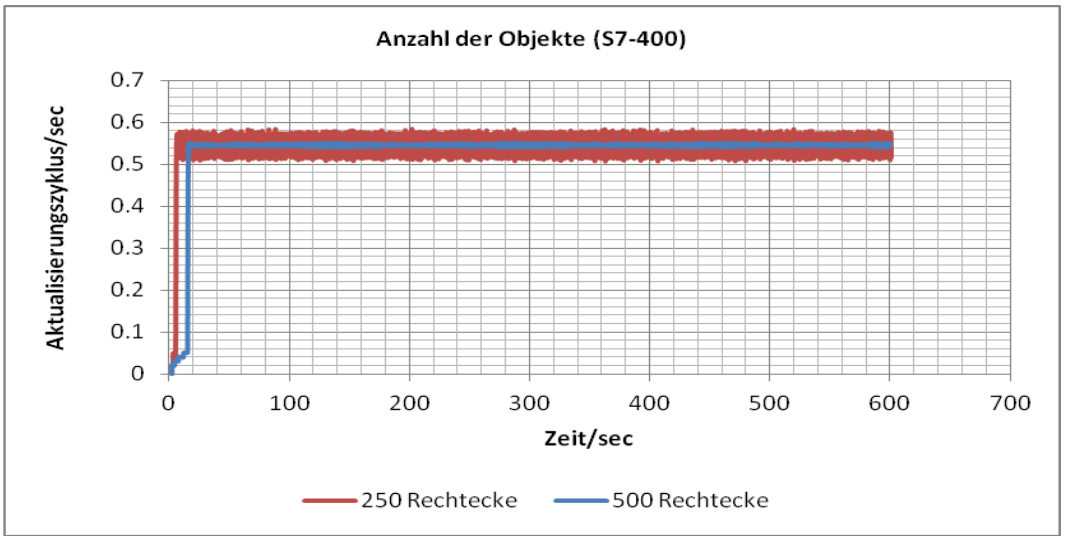


Diagramm 5-13: Rechteck im Bild mit 250 und 500 Objekte (S7-400 bei zyklischen Trigger)



Von den obigen Diagrammen kann man sehen:

Wenn 250 Objekte mit Variablentrigger projektiert werden, ist die Aktualisierungzeit bei S7-300 ein wenig kleiner als die bei 500 Objekten (Diag.5-10). Bei S7-400 konnte keinen Unterschied bei der Aktualisierungzeit festgestellt werden (Diag.5-11).

Wenn 250 Objekte mit zyklischem Trigger projektiert werden, sind die Reaktionszeit und die tatsächliche Aktualisierungszeit bei S7-300 kleiner als die bei 500 Objekten (Diag.5-12).

Bei S7-400 ist die Reaktionszeit bei 250 Objekte mit zyklischem Trigger kleiner als bei 500 Objekten. Die Aktualisierungszeit liegt aber im Intervall von 0,5 bis 0,57 sec (Diag. 5-13).

Je mehr mit den Prozessvariablen dynamisierten Objekten im Bild, desto mehr Telegramme werden ständig zwischen WinCC und Steuerung ausgetauscht. Dies hat dementsprechend Einfluss auf die Kommunikationslast.

5.2.3.5 Belastung von Tag Logging und Alarm Logging

Messumgebung:

Tabelle 5-23: Einstellung von WinCC, Wireshark zum Test: „Belastung von Tag Logging und Alarm Logging“

WinCC	Verbindungsart	Lesedienstart	**Testbild**	Aktualisierungszyklus	**Logging**
	TCP/IP	Pollen	**TB4, TB7**	1000ms	**aktiv / deaktiv**
Wireshark	Messintervall	Anweisung zum Greifen der Telegramme			
	600sec	**Bei Logging: deaktiv** **[7]frame[-5:5]==15:84:00:05:00** bei S7-300 für TB4 **[8]frame[-5:5]==15:84:00:05:20** bei S7-300 für TB7 **[11]frame[-5:5]==1c:00:17:00:20** bei S7-400 für TB4 **[12]frame[-5:5]==20:00:17:00:20** bei S7-400 für TB7 **Bei Logging: deaktiv siehe die Anweisungen im 5.2.3.2**			

Messergebnis:

Diagramm 5-14: „Logging aktiv / deaktiv“ (S7-300 bei zyklischen Trigger)

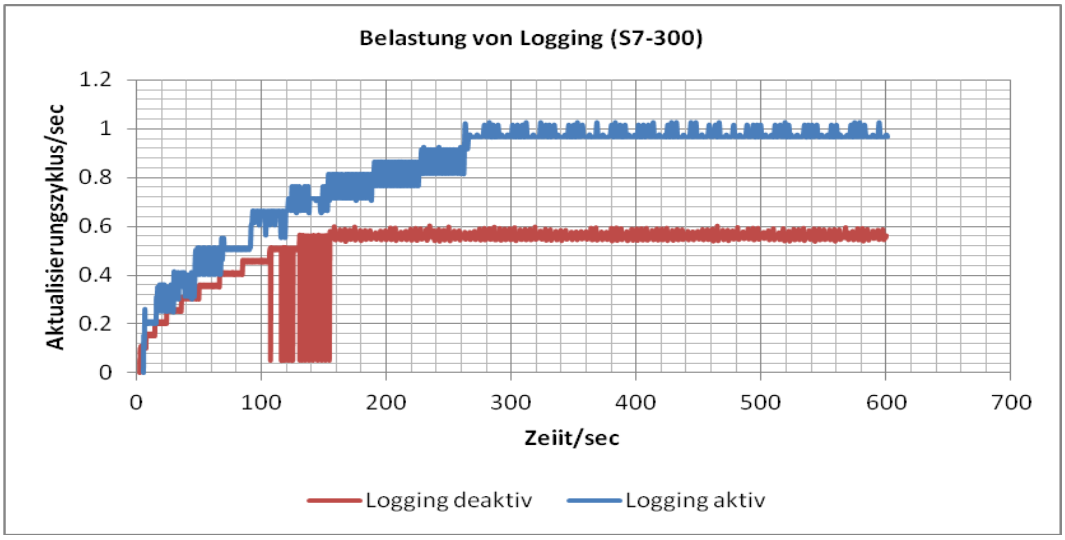


Diagramm 5-15: „Logging aktiv / deaktiv“ (S7-400 bei zyklischen Trigger)

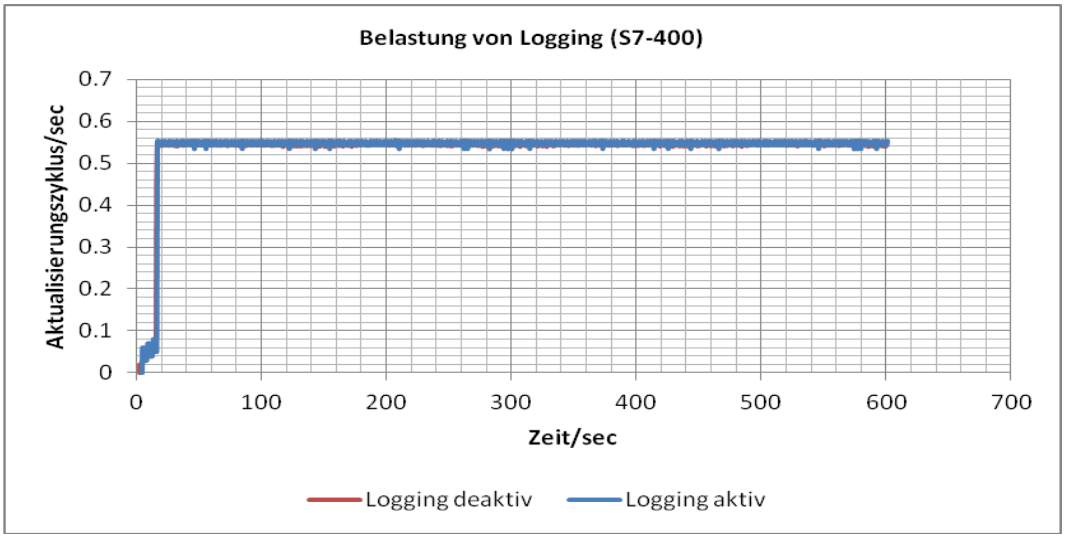


Diagramm 5-16: „Logging aktiv / deaktiv“ (S7-300 bei Variablentrigger)

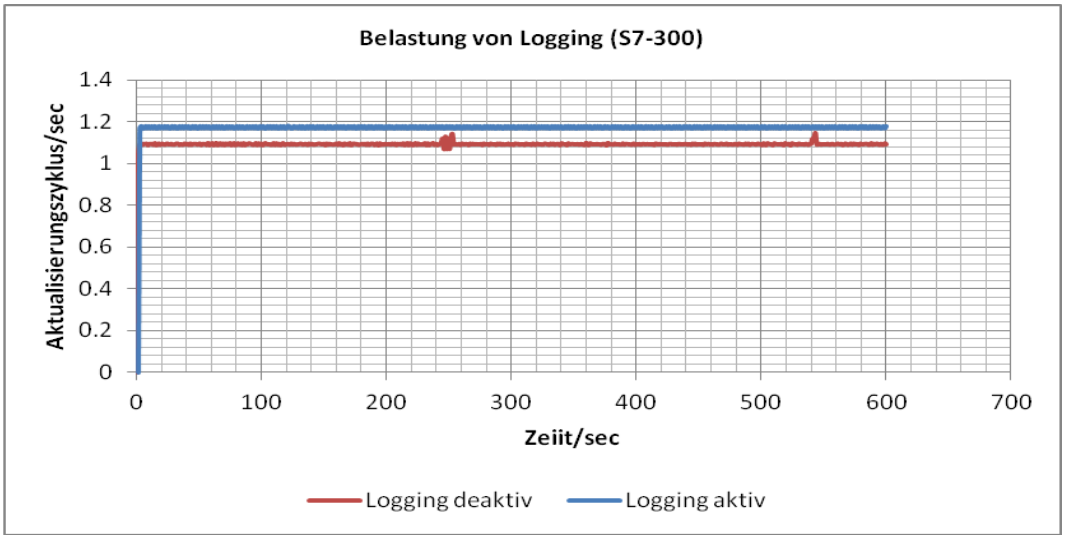
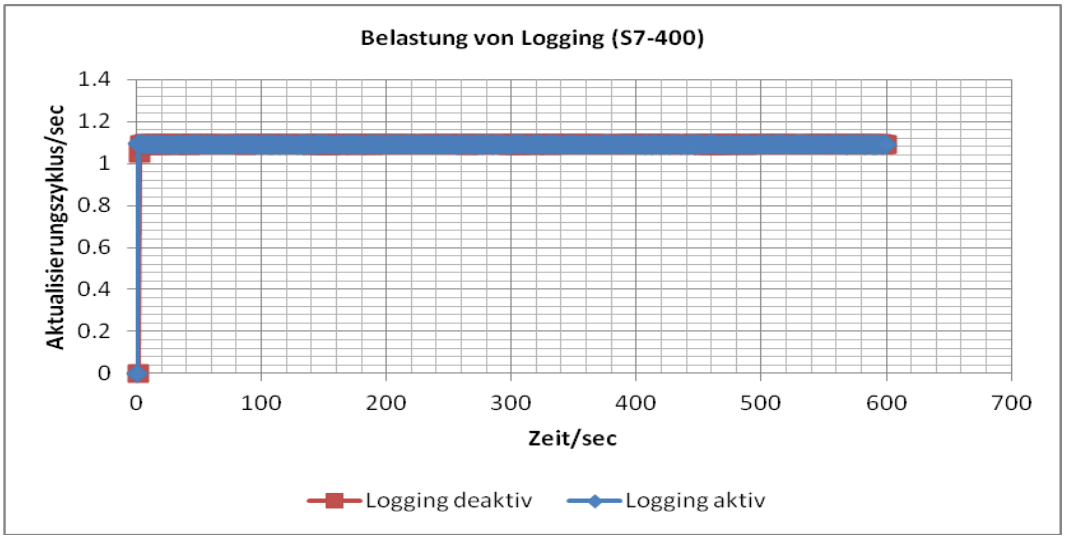


Diagramm 5-17: „Logging aktiv / deaktiv“ (S7-400 bei Variablentrigger)



Die obigen Diagramme zeigen:
Bei S7-300: Wenn bei WinCC Server Tag Logging und Alarm Logging aktiviert sind, wird die Kommunikationslast erheblich erhöht. Dadurch wird die tatsächliche Aktualisierungszeit im Bild verschlechtert. Diese ist unabhängig von den gewählten Triggerart.
Die Ursache ist wie im letzten Test erklärt ersichtlich. Durch die Aktivierung von Alarm Logging und Tag Logging werden zusätzlich Variablen an den Datenmanager angemeldet und Anzahl der Telegramme zwischen WinCC und Steuerung wird dementsprechend erhöht, was wiederum zum mehr Kommunikationslast führt.
Bei S7-400: Die Belastung von „Tag Logging“ und „Alarm Logging“ hat keinen großen Einfluss auf die Kommunikationsperformance in diesem Test.

5.2.4 Ergänzung der Untersuchung von „Lesedienstart“

Messumgebung:

Tabelle 5-24: Allgemeine Einstellung der Messumgebung zum Test: „Ergänzung der Untersuchung von Lesedienstart“

Steuerung	OB1 Zyklus	Priorisierte BuB-Kommunikation	Zyklusbelastung durch Kommunikation		
	50ms	deaktiv	20%		
WinCC	Verbindungsart	**Lesedienstart**	Testbild	Aktualisierungszyklus	Logging
	TCP/IP	**Pollen, zyklisches Lesen, zyklisches Lesen bei Änderung**	TB4	100ms	deaktiv
Wireshark	Messintervall	Anweisung zum Greifen der Telegramme			
	600sec	[1]frame[-5:5]==15:84:00:05:00 bei S7-300 [4]frame[-5:5]==20:00:16:00:00 bei S7-400			

Messergebnis:

Diagramm 5-18: Aufträge von WinCC bei verschiedenen Lesediensten (S7-400)

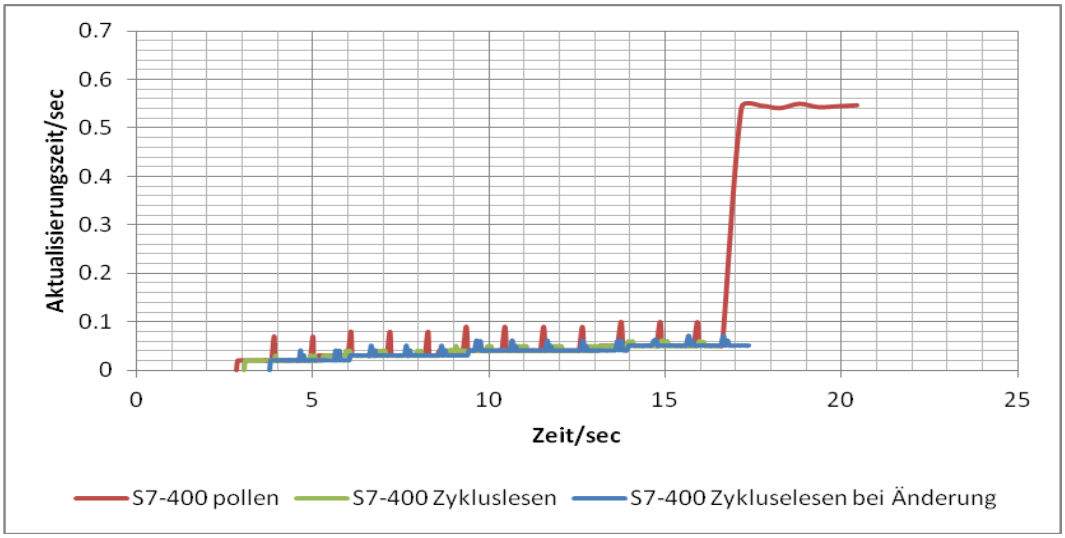
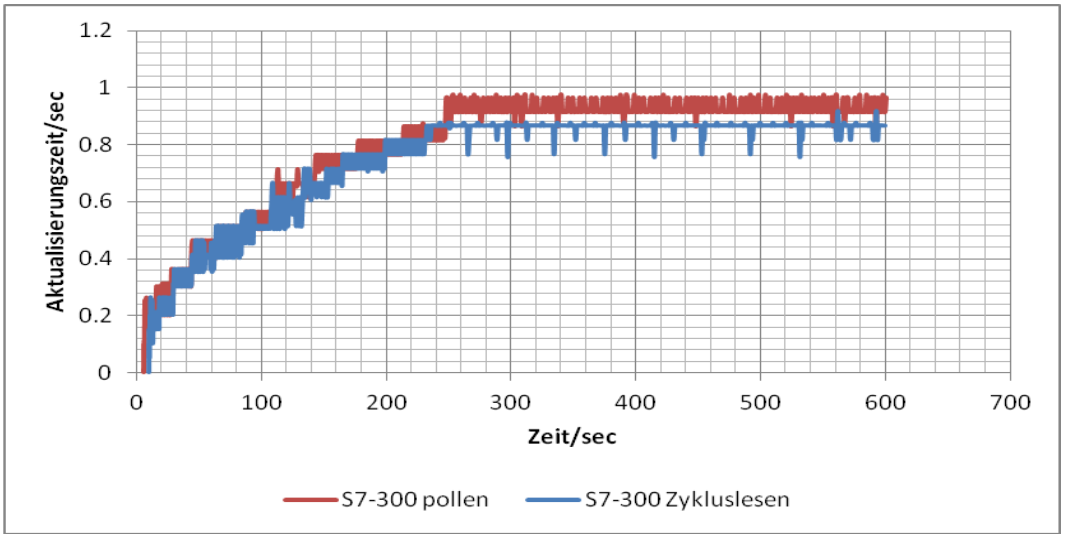


Diagramm 5-19: Aufträge von WinCC bei verscheidenen Lesediensten (S7-300)



Aus Diag. 5-18 kann man sehen, dass das Verhältnis und die Tendenz der Aufträge eindeutig wie die Resultante im Abschnitt 5.2.1 sind. Aber die Tendenz vom Telegramm bei S7-300, Lesedienstart. „Zykluslesen“ verhält sich nicht wie erwartet. Das Prinzip beim „zyklischen Lesedienst“ sollte so sein, dass nachdem alle Variablen nach Bildanwahl von WinCC angefordert sind, soll es keinen weiteren Leseauftrag von WinCC an die Steuerung gesendet werden. Aber im Diag.5-19 kann man feststellen, dass die Leseaufträge auch nach der Reaktionszeit noch existieren.
Die mögliche Ursache wäre die Beschränkung der Anzahl von der Leseaufträge bei S7-300.

Um die genaue Ursache herauszufinden, fügen wir nach einander 46, 92, 100 Rechtecke (mit 46, 92, 100 DWORD Prozessvariablen verbunden) im Testbild ein. Die andere Messumgebung ändert sich nicht.

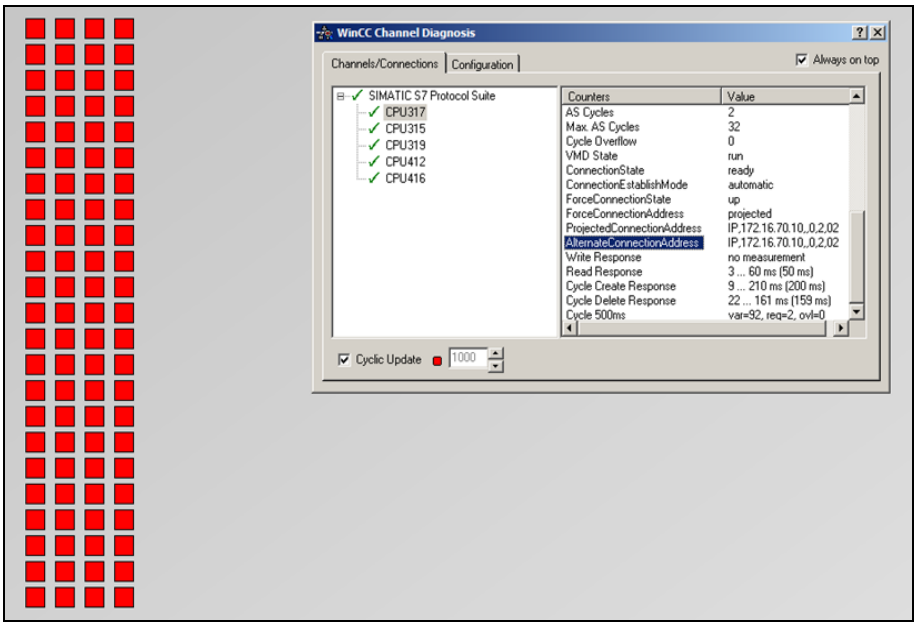


Abbildung 5-16: Testbild mit 92 Rechtecke (mit 92 Variablen verbunden)

Counters	Value
AS Cycles	2
Max. AS Cycles	32
Cycle Overflow	0
VMD State	run
ConnectionState	ready
ConnectionEstablishMode	automatic
ForceConnectionState	up
ForceConnectionAddress	projected
ProjectedConnectionAddress	IP,172.16.70.10,,0,2,02
AlternateConnectionAddress	IP,172.16.70.10,,0,2,02
Write Response	no measurement
Read Response	3 ... 60 ms (50 ms)
Cycle Create Response	9 ... 210 ms (200 ms)
Cycle Delete Response	22 ... 161 ms (159 ms)
Cycle 500ms	var=92, req=2, ovl=0

Abbildung 5-17: WinCC Kanal Diagnose (bei 92 Prozessvariable, 2 Aufträge)

No.	Time	Delta Time display	Source	Destination	Protocol	Length	Info
642	7.249348000	0.002826000	172.16.70.10	172.16.70.55	T.125	271	detachUserRequest
644	7.257391000	0.008043000	172.16.70.55	172.16.70.10	T.125	145	detachUserRequest
645	7.260349000	0.002958000	172.16.70.10	172.16.70.55	T.125	283	detachUserRequest
647	7.266509000	0.006160000	172.16.70.55	172.16.70.10	T.125	145	detachUserRequest
648	7.269345000	0.002836000	172.16.70.10	172.16.70.55	T.125	275	detachUserRequest
650	7.278019000	0.008674000	172.16.70.55	172.16.70.10	T.125	145	detachUserRequest
651	7.280358000	0.002339000	172.16.70.10	172.16.70.55	T.125	283	detachUserRequest
653	7.286634000	0.006276000	172.16.70.55	172.16.70.10	T.125	145	detachUserRequest
654	7.289366000	0.002732000	172.16.70.10	172.16.70.55	T.125	279	detachUserRequest
656	7.298083000	0.008717000	172.16.70.55	172.16.70.10	T.125	145	detachUserRequest
657	7.300354000	0.002271000	172.16.70.10	172.16.70.55	T.125	283	detachUserRequest
659	7.306814000	0.006460000	172.16.70.55	172.16.70.10	T.125	145	detachUserRequest
660	7.309353000	0.002539000	172.16.70.10	172.16.70.55	T.125	283	detachUserRequest
662	7.748204000	0.438851000	172.16.70.55	172.16.70.10	T.125	159	detachUserRequest
664	7.835414000	0.087210000	172.16.70.10	172.16.70.55	T.125	297	detachUserRequest
666	7.853292000	0.017878000	172.16.70.55	172.16.70.10	T.125	159	detachUserRequest
668	7.937447000	0.084155000	172.16.70.10	172.16.70.55	T.125	297	detachUserRequest
676	8.295252000	0.357805000	172.16.70.10	172.16.70.55	T.125	297	detachUserRequest
678	8.396413000	0.101161000	172.16.70.10	172.16.70.55	T.125	297	detachUserRequest
680	8.804419000	0.408006000	172.16.70.10	172.16.70.55	T.125	297	detachUserRequest
682	8.906491000	0.102072000	172.16.70.10	172.16.70.55	T.125	297	detachUserRequest
690	9.314501000	0.408010000	172.16.70.10	172.16.70.55	T.125	297	detachUserRequest
692	9.416475000	0.101974000	172.16.70.10	172.16.70.55	T.125	297	detachUserRequest
694	9.824487000	0.408012000	172.16.70.10	172.16.70.55	T.125	297	detachUserRequest

Abbildung 5-18: Telegramme bei „Zykluslesen“, 2 Aufträge

Durch den Vergleich von Abbildungen (Ab.5-17 und Ab.5-18) stellt man fest, dass der „Zykluslesen“ bei S7-300 bis zu 2 Telegramm (Aufträge) richtig funktioniert.

Counters	Value
AS Cycles	2
Max. AS Cycles	32
Cycle Overflow	0
VMD State	run
ConnectionState	ready
ConnectionEstablishMode	automatic
ForceConnectionState	up
ForceConnectionAddress	projected
ProjectedConnectionAddress	IP,172.16.70.10,,0,2,02
AlternateConnectionAddress	IP,172.16.70.10,,0,2,02
Write Response	no measurement
Read Response	3 ... 13 ms (60 ms)
Cycle Create Response	(175 ms)
Cycle Delete Response	(161 ms)
Cycle 500ms	var=100, req=3, ovl=0

Abbildung 5-19: WinCC Kanal Diagnose (bei 100 Prozessvariablen, 3 Aufträge)

No.	Time	Delta Time display	Source	Destination	Protocol	Length	Info
459	2.933837000	0.007325000	172.16.70.55	172.16.70.10	T.125	85	detachUserRequest
460	2.936425000	0.002588000	172.16.70.10	172.16.70.55	T.125	111	detachUserRequest
468	3.463612000	0.527187000	172.16.70.55	172.16.70.10	T.125	159	detachUserRequest
470	3.534572000	0.070960000	172.16.70.10	172.16.70.55	T.125	297	detachUserRequest
472	3.538361000	0.003789000	172.16.70.55	172.16.70.10	T.125	159	detachUserRequest
474	3.636573000	0.098212000	172.16.70.10	172.16.70.55	T.125	297	detachUserRequest
476	3.639111000	0.002538000	172.16.70.55	172.16.70.10	T.125	99	detachUserRequest
477	3.641424000	0.002313000	172.16.70.10	172.16.70.55	T.125	87	detachUserRequest
479	3.649301000	0.007877000	172.16.70.55	172.16.70.10	T.125	85	detachUserRequest
480	3.651466000	0.002165000	172.16.70.10	172.16.70.55	T.125	111	detachUserRequest
482	4.044494000	0.393028000	172.16.70.10	172.16.70.55	T.125	297	detachUserRequest
484	4.095622000	0.051128000	172.16.70.10	172.16.70.55	T.125	297	detachUserRequest
486	4.118562000	0.022940000	172.16.70.55	172.16.70.10	T.125	85	detachUserRequest
487	4.121533000	0.002971000	172.16.70.10	172.16.70.55	T.125	111	detachUserRequest
495	4.503655000	0.382122000	172.16.70.10	172.16.70.55	T.125	297	detachUserRequest
497	4.605638000	0.101983000	172.16.70.10	172.16.70.55	T.125	297	detachUserRequest
499	4.666572000	0.060934000	172.16.70.55	172.16.70.10	T.125	85	detachUserRequest
500	4.669550000	0.002978000	172.16.70.10	172.16.70.55	T.125	111	detachUserRequest
502	5.013580000	0.344030000	172.16.70.10	172.16.70.55	T.125	297	detachUserRequest
504	5.115667000	0.102087000	172.16.70.10	172.16.70.55	T.125	297	detachUserRequest

Abbildung 5-20: Telegramme bei „Zykluslesen“, 3 Aufträge

Die obigen 2 Abbildungen zeigen, dass wenn die Anzahl der Aufträge (Telegramme) über 2 ist, ist „Zykluslesen“ für weiteren Variablen nicht mehr gültig. Die Variablen ab 3ten Auftrag (Telegramme) werden durch „Pollen“ an der Steuerung gesendet.

Es dürfen max. 2 zyklische Lesedienste pro Verbindung bei S7-300 bzw. bis 32 zyklische Lesedienste pro Verbindung bei S7-400 verwendet werden (CPU abhängig). Erstreckt sich die angeforderten Daten über 2 Aufträge, werden nur dann die ersten beiden Aufträge als „zyklischen Lesedienst“ bei der Steuerung angemeldet. Die weiteren Aufträge werden durch „azyklischen Lesedienst“ (Pollen) an die Steuerung gesendet.
In einem Telegramm bei S7-300 können max. 46 Variablen vom Typ DWORD verpackt werden, bei S7-400 ist es max. 108 Variablen (DWORD). Also ab 93. Variable muss WinCC diese Daten dann über „azyklische Leseaufträge“ anfordern und die Zyklusbildung selbst vornehmen.
Bei S7-400 ist ab Variable (DWORD) 1729 (S7-412: 16 Verbindungen x 108 Tags = 1728) bzw. bei S7-417 ab Variable 3457 (32 Verbindungen x 108 Tags = 3456) werden durch „Pollen“ gelesen.

5.2.5 Die Messaufgaben mit dem Protokoll: PROFIBUS

In dieser Messaufgabe werden die Parameter bei der Verbindung mit PROFIBUS ermittelt. Weil für das Profibus Wirshark nicht eingesetzt werden kann, wird der Test mit Hilfe von manuellen Zeitmessungen, und WinCC-Diagnosetool durchgeführt. Ich wähle bei dieser Messaufbabe einige Einflussfaktoren. Die Ergebnisse werden dann mit der TCP/IP verglichen.

Zur Test: 5.2.2 Einstellung der Steuerung

<u>Messumgebung:</u>

Tabelle 5-25: Einstellung von Steuerung und WinCC bei PROFIBUS (1)

Steuerung	OB1 Zyklus	Priorisierte BuB-Kommunikation	Zyklusbelastung durch Kommunikation		
	50ms	aktiv	20%		
WinCC	**Verbindungsart**	Lesedienstart	Testbild	Aktualisierungszyklus	Logging
	PROFIBUS	Pollen	TB2	100ms	deaktiv
Messtool					
manuell Messen mit Zeitmesser					

Messergebnis:

Tabelle 5-26: Zugriffszeit bei PROFIBUS und TCP/IP

	PROFIBUS		**TCP/IP**	
	317 2PN/DP	412 2PN	317 2PN/DP	412 2PN
Einheit: [s]	500 Variablen von DB31Var0 bis DB32Var249			
Manuell	5.0	2.5	5.0	2.5
Einheit [s]	1000 Variablen von DB31Var0 bis DB34Var249			
Manuell	-	5.0		5.0

Zur Test: 5.2.3 Auswirkung auf Bildaufbau und Konfiguration von WinCC

Messumgebung:

Tabelle 5-27: Einstellung von Steuerung und WinCC bei PROFIBUS (2)

Steuerung	OB1 Zyklus	Priorisierte BuB-Kommunikation	Zyklusbelastung durch Kommunikation		
	50ms	deaktiv	20%		
WinCC	**Verbindungsart**	Lesedienstart	**Testbild**	Aktualisierungszyklus	Logging
	PROFIBUS	Pollen	**TB4, TB10**	100ms	deaktiv
Messtool			Messintervall		
manuell Messen mit Zeitmesser			600sec		

Messergebnis:

Tabelle 5-28: Reaktionszeit bei PROFIBUS und TCP/IP

	PROFIBUS		**TCP/IP**	
	317 2PN/DP	412 2PN	317 2PN/DP	412 2PN
Einheit: [s]	**bei TB4**			
Manuell	156	15	155	15
Einheit: [s]	**bei TB10**			
Manuell	245	16	245	15

Aus der obigen Tabelle kann man sehen:

Es wurde keinen signifikanten Unterschied bezüglich Performancelast zwischen PROFIBUS und TCP/IP in diesem Test festgestellt. Eine mögliche Ursache wäre, dass in diesem Test die Kommunikationslast nicht sehr groß war.

5.2.6 Vergleich der Performance bei Bilderaktualisierung am WinCC Client und WinCC Server

In dieser Messaufgabe werden die Parameter am WinCC Client gemessen, dann werden die gemessene Aktualisierungen am WinCC Client mit der von WinCC Server verglichen. Die andere Messumgebung ist gleich wie die bei Abschnitt 5.2.5 (siehe Tab.5-27)

Messergebnis:

Tabelle 5-29: Reaktionszeit bei WinCC Server und WinCC Client (TCP/IP)

	WinCC Server		**Win CC Client**	
	317 2PN/DP	412 2PN	317 2PN/DP	412 2PN
Einheit: [s]		**bei TB4**		
Manuell	155	15	161	26
Einheit: [s]		**bei TB10**		
Manuell	245	15	249	29

Tabelle 5-30: Reaktionszeit bei WinCC Server und WinCC Client (PROFIBUS)

	WinCC Server		**Win CC Client**	
	317 2PN/DP	412 2PN	317 2PN/DP	412 2PN
Einheit: [s]		**bei TB4**		
Manuell	156	15	161	27
Einheit: [s]		**bei TB10**		
Manuell	245	16	249	29

Aus den Daten in den obigen Tabellen kann eine geringfügige (ca. 2%) Aktualisierungszeit am Client und Server bei S7-317 CPU beobachtet werden. Wohingegen ist die Aktualisierungszeit der Bilder am Client und Server bei S7-412 erheblich höher (ca. 80%). Es konnte keine Erklärung für diesen Effekt bei S7-412 ermittelt werden.

6 Fazit und Ausblick

Dieses Kapitel fasst abschließend die Resultate der Bachelorarbeit zusammen. Zusätzlich wird ein Ausblick auf mögliche Weiterentwicklungen gegeben.

6.1 Fazit/Zusammenfassung

Aus den durchgeführten Messaufgaben können folgende Aussagen bezüglich der Kommunikationslast gemacht werden.

1. Auf der Steuerungsseite sind die „Einstellungen der Parameter“ von der CPU, „Belastung des SPS-Programms“ und der „Typ“ der Steuerung Haupteinflussfaktoren. Die S7-400 zeigt viel bessere Leistung als S7-300 für die Kommunikation mit WinCC, da sie 2 Prozessoren hat und dadurch mehr Ressourcen für die Kommunikation zur Verfügung stellen kann.
Die Funktion: „priorisierte BuB Kommunikation“ bei S7-300 kann die Performance der Kommunikation mit WinCC erheblich verbessern. Allerdings ist die Konsistenz der Daten zum Anwenderprogramm nicht mehr 100% gewährleistet. Dies muss bei solchen Lösungen berücksichtigt werden.

2. Die Einstellung des Lesedienstes hat auch einen großen Einfluss auf die Kommunikationsperformance. Die Einstellung „zyklische Lesedienste“ kann bei WinCC eingestellt werden. Dadurch wird die Kommunikationslast erheblich verbessert. Allerdings wird der Zyklische Lesedeinst nur von bestimmten Typen von S7-300 CPUs unterstützt.
Das AS und die AS-OS-Kommunikation wird durch zyklische weniger belastet, da nicht ständig Leseaufträge an das AS gesendet und dort bearbeitet werden müssen.
S7-300 unterstützt maximal 2 zyklische Lesedienste, S7-400 je nach CPU-Typ 16 (S7-412) bzw. 32 (S7-416) zyklische Lesedienste.
Bei 400 CPUs kann zusätzlich die „Änderungsübertragung“ eingestellt werden. Dadurch werden nur die geänderten Werte an WinCC gesendet.

3. Die Haupteinflussfaktoren sind die Art der Projektierung der Bilder (Dynamisierungen und Trigger), Alarm- und Tag Logging, Globale Aktionen, … in WinCC und die Größe des WinCC-Projektes.
Je mehr Prozessvariablen und je häufiger die Daten WinCC fordert, desto mehr Aufträge müssen zwischen dem Datenmanager und Steuerung ausgetauscht werden und dies führt wiederum zu mehr Kommunikationslast.

4. Die Bildaktualisierung am WinCC Server ist schneller als am WinCC Client.

Die Kommunikationsperformance von WinCC ist abhängig von der kompletten Konfiguration der Steuerung und WinCC. Die Ressourcen für Kommunikation sind begrenzt. Die gemessenen Einflussfaktoren in dieser Bachelorarbeit liefert eine Abhilfe, um eine flexible Ressourcenallokation und Optimierung der Kommunikation zwischen WinCC und Steuerung bei der Anwendung zu realisieren.

6.2 Ausblick

Folgende Weiterentwicklungen des Messprojektes sind denkbar:

Aus Zeitgründen konnte nur die Kommunikationsperformance von TCP/IP Protokoll detalliert untersucht werden. Weitere Kommunikationsarten konnten nicht näher untersucht werden. Deshalb konnte keinen weiteren Vergleich zwischen verschiedenen Protokollarten ausgearbeitet werden.

Die Kommunikationsresourcen von CPUs konnten nicht durch das erstellte Simulationsprogramm ausgeschöpft werden. Deshalb war es nicht möglich den Fall, bei dem keine bzw. nicht genügend zur Verfügung stehende Kommunikatiosresourcen näher untersucht werden.

Es wäre interessant, wenn weitere Konfigurationen noch untersucht werden könnten:
1. Gleichzeitige Kommuikation von mehreren WinCC-Servern zu einer Steuerung
2. Redundante WinCC-Server
3. Server/Client Konfiguration mit mehr Client-Anbindung (bis zu 32 Clients)

Literatur- und Quellenverzeichnis

[1] Siemens: Produkte für Totally Integrated Automation und Micro Automation Katalog ST 70.2011

[2] Siemens: Bedien- und Beobachtungssysteme/PC-based Automation Katalog ST 80/ST PC:2012

[3] Siemens: WinCC Communication Manual Handbuch 1999 BestellNr. 6AV6392-1CA05-0AA0

[4] Siemens: WinCC Configuration Manual Handbuch 1999 BestellNr. 6AV6392-1CA05-0AA0

[5] Siemens: WinCC V7.0 SP1 MDM- WinCC: Kommunikation Systemhandbuch 11/2008

[6] Siemens: WinCC V7.0 SP1 MDM- WinCC: Scripting (VBS, ANSI-C, VBA) Systemhandbuch 11/2008

[7] Siemens: S7-300 CPU 31xC und CPU 31x: Technische Daten Gerätehandbuch 08/2010 BestellNr. 6ES7398-8FA10-8AA0

[8] Siemens: S7-400 Automatisierungssystem S7-400 CPU-Daten Gerätenhandbuch 08/2010 BestellNr. 6ES7498-8AA05-8AA0

[9] Siemens: SIMATIC NET Industrial Ethernet Switches SCALANCE X-200 Projektierungshandbuch 06/2012

[10] Siemens: Industry Online Support
Weblink:http://support.automation.siemens.com/WW/llisapi.dll?func=cslib.csinfo2&siteid=cseus&lang=de&aktprim=99&siteid=cseus&extranet=standard&viewreg=WW&groupid=4000003

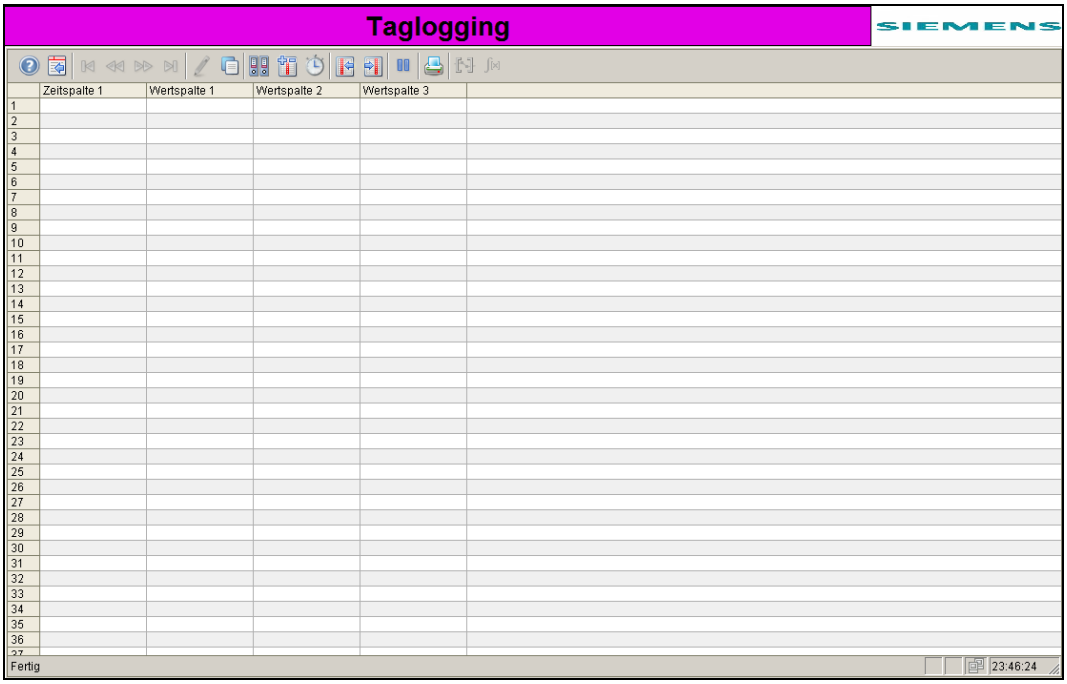
[11] Wireshark: Wireshark User's Guide for Wireshark 1.9
Weblink: http://www.wireshark.org/docs/wsug_html_chunked/

Anhang

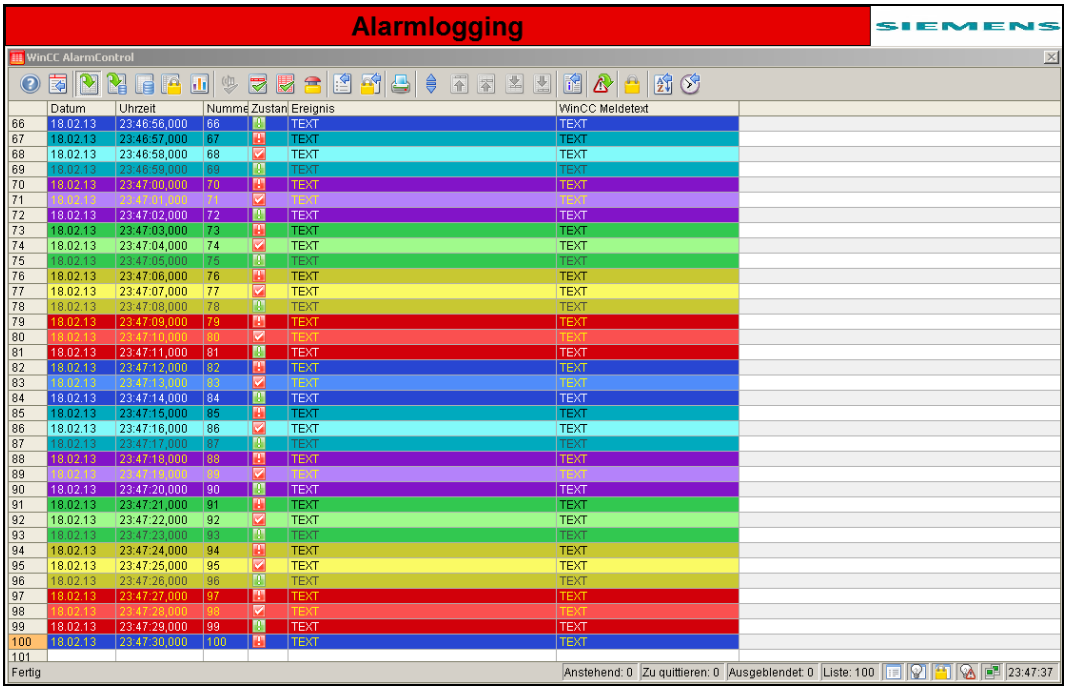
A [01] Datei des Prozessbildaufbaus

Bildnummer	Bildname	Objekt				
KB1	„Tag Logging“	WinCC Online Table Control				
KB2	„Alarm Logging“	WinCC Alarm Control				
Bildnummer	Bildname	Objekt ID	Anzahl	Variable		
TB1	„Lesedienst“	Obj.1	1	Var.0 von DB31		
Bildnummer	Bildname	Objekt ID	Anzahl	Variable		
TB2	„Belastungstest“	Obj.1	500	Variablen von DB31-DB32		
TB3	„Belastungstest 4DB“	Obj.1	1000	Variablen von DB31-DB34		
Bildnummer	Bildname		Objekt ID	Anzahl	Variable	Triggerart
TB4	„Zeittrigger mit C Rechteck“		Obj.3	500	DB21-DB25	rein
TB5	„Zeittrigger mit VBS Rechteck“		Obj.4	500	DB21-DB25	rein
TB6	„Zeittrigger mit DynamikDialog Rechteck“		Obj.5	500	DB21-DB25	rein
TB7	„Variablentrigger mit C Rechteck“		Obj.6	500	DB21-DB25	rein
TB8	„Variablentrigger mit VBS Rechteck“		Obj.7	500	DB26-DB30	rein
TB9	„Variablentrigger mit DynamikDialog Rechteck“		Obj.8	500	DB26-DB30	rein
TB10	„Zeittrigger mit C“		Obj.3	500	DB21-DB25	gemischt
			Obj.2	500	DB26-DB30	
TB11	„Zeittrigger mit VBS“		Obj.4	500	DB21-DB25	gemischt
			Obj.2	500	DB26-DB30	
TB12	„Zeittrigger mit DynamikDialog“		Obj.5	500	DB21-DB25	gemischt
			Obj.2	500	DB26-DB30	
TB13	„Variablentrigger mit C“		Obj.6	500	DB21-DB25	rein
			Obj.2	500	DB26-DB30	
TB14	„Variablentrigger mit VBS“		Obj.7	500	DB21-DB25	rein
			Obj.2	500	DB26-DB30	
TB15	„Variablentrigger mit DynamikDialog“		Obj.8	500	DB21-DB25	rein
			Obj.2	500	DB26-DB30	
TB16	„Zeittrigger mit C 500“		Obj.3	250	DB21-DB25	gemischt
			Obj.2	250	DB26-DB30	

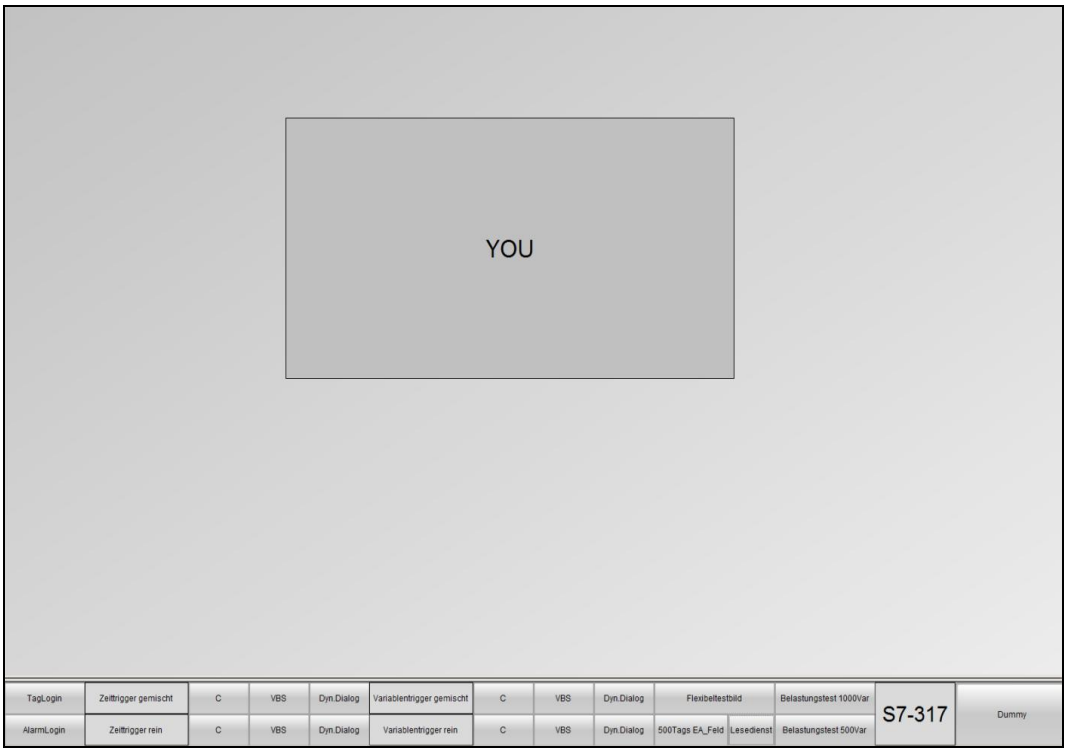
A [02] Abbildungen des Prozessbilder



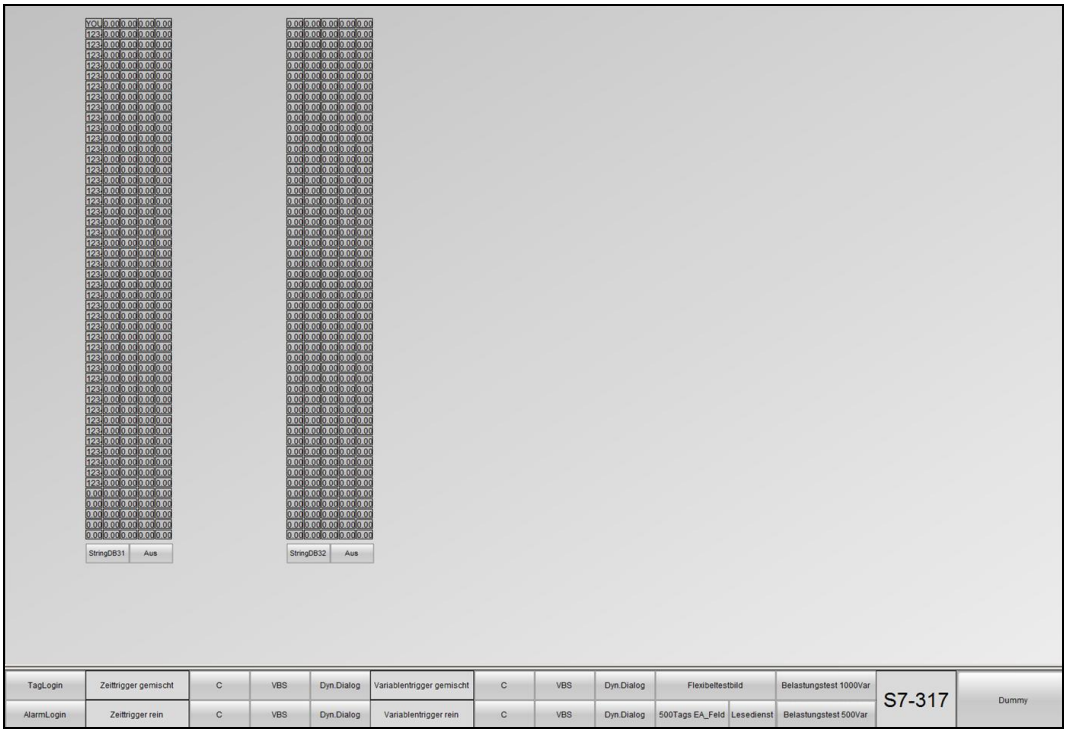
KB1



KB2

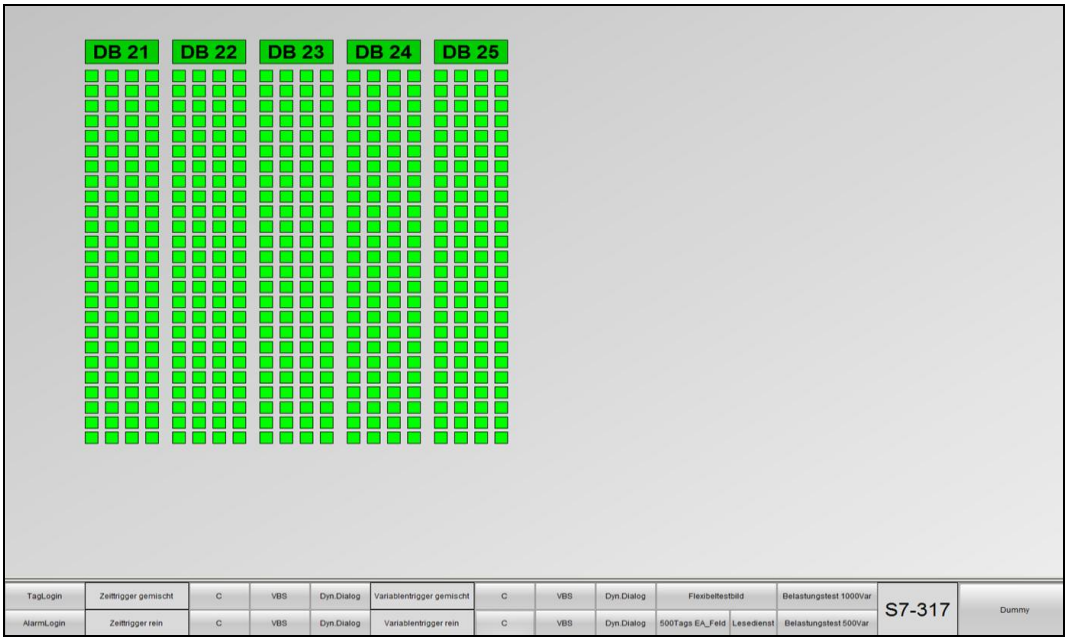

YOU
TagLogin
Zeittrigger gemischt
C
VBS
Dyn.Dialog
Variablentrigger gemischt
C
VBS
Dyn.Dialog
Flexibeltestbild
Belastungstest 1000Var
S7-317
Dummy
AlarmLogin
Zeittrigger rein
C
VBS
Dyn.Dialog
Variablentrigger rein
C
VBS
Dyn.Dialog
500Tags EA_Feld
Lesedienst
Belastungstest 500Var

TB1


YOU
StringDB31
Aus
StringDB32
Aus
TagLogin
Zeittrigger gemischt
C
VBS
Dyn.Dialog
Variablentrigger gemischt
C
VBS
Dyn.Dialog
Flexibeltestbild
Belastungstest 1000Var
S7-317
Dummy
AlarmLogin
Zeittrigger rein
C
VBS
Dyn.Dialog
Variablentrigger rein
C
VBS
Dyn.Dialog
500Tags EA_Feld
Lesedienst
Belastungstest 500Var

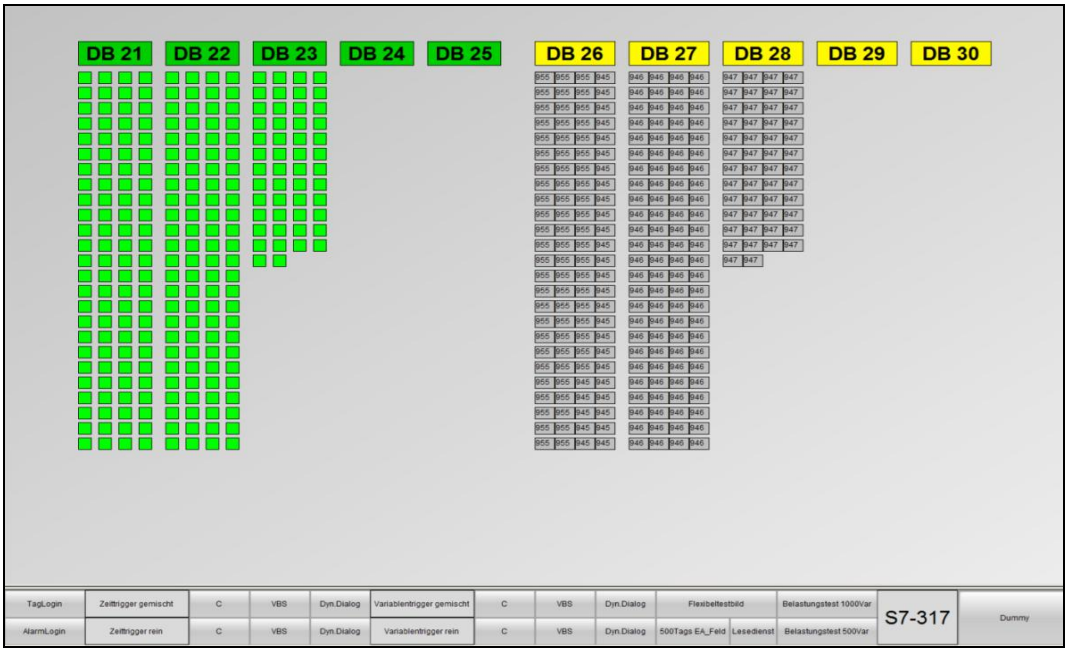
TB2

TB3



TB4, TB5, TB6, TB7, TB8, TB9

TB10, TB11, TB12; TB13, TB14, TB15



TB16

A [03] Anweisungen zum Greifen des Telegramms

Tag Logging, Alarm Logging deaktiv

Objekt: 500 Rechtecke

	TB4, TB5, TB6	TB7, TB8, TB9
S7 300	[1]frame[-5:5]==15:84:00:05:00	[1]frame[-5:5]==15:84:00:05:00
S7 400	[4]frame[-5:5]==20:00:16:00:00	[4]frame[-5:5]==20:00:16:00:00

Objekt: 500 Rechtecke + 500 EA_Felder

	TB10, TB11, TB12	TB13, TB14, TB15
Rechteck		
S7 300	[1]frame[-5:5]==15:84:00:05:00	[1]frame[-5:5]==15:84:00:05:00
S7 400	[4]frame[-5:5]==20:00:16:00:00	[4]frame[-5:5]==20:00:16:00:00
EA_Feld		
S7 300	[2]frame[-5:5]==1a:84:00:05:00	[3]frame[-5:5]==1a:84:00:05:c0
S7 400	[5]frame[-5:5]==20:00:1b:00:00	[6]frame[-5:5]==20:00:1b:00:a0

S7-300

[1]frame[-5:5]==15:84:00:05:00

[2]frame[-5:5]==1a:84:00:05:00

[3]frame[-5:5]==1a:84:00:05:c0

S7-400

[4]frame[-5:5]==20:00:16:00:00

[5]frame[-5:5]==20:00:1b:00:00

[6]frame[-5:5]==20:00:1b:00:a0

Tag Logging, Alarm Logging aktiv

Objekt: 500 Rechtecke

	TB4, TB5, TB6	TB7, TB8, TB9
S7 300	[7]frame[-5:5]==15:84:00:05:00	[8]frame[-5:5]==15:84:00:05:20
S7 400	[12]frame[-5:5]==20:00:17:00:20	[11]frame[-5:5]==1c:00:17:00:20

Objekt: 500 Rechtecke + 500 EA_Felder

	TB10, TB11, TB12	TB13, TB14, TB15
Rechteck		
S7 300	[7]frame[-5:5]==15:84:00:05:00	[8]frame[-5:5]==15:84:00:05:20
S7 400	[12]frame[-5:5]==20:00:17:00:20	[11]frame[-5:5]==1c:00:17:00:20
EA_Feld		
S7 300	[9]frame[-5:5]==1a:84:00:05:20	[10]frame[-5:5]==1a:84:00:05:e0
S7 400	[13]frame[-5:5]==1c:00:1c:00:20	[14]frame[-5:5]==1c:00:1c:00:c0

S7-300

[7]frame[-5:5]==15:84:00:05:00

[8]frame[-5:5]==15:84:00:05:20

[9]frame[-5:5]==1a:84:00:05:20

[10]frame[-5:5]==1a:84:00:05:e0

S7-400

[11]frame[-5:5]==1c:00:17:00:20

[12]frame[-5:5]==20:00:17:00:20

[13]frame[-5:5]==1c:00:1c:00:20

[14]frame[-5:5]==1c:00:1c:00:c0

A [04] Die ausgewählte Telegramme

S7-300 Tag Logging, Alarm Logging deaktiv

```
0030   0b 68 d1 78 00 00 03 00   00 5b 02 f0 80 32 01 00   .h.x.... .[...2..
0040   00 5c 00 00 4a 00 00 04   06 12 0a 10 02 00 20 00   .¥..J... ...... .
0050   15 84 00 00 00 12 0a 10   02 00 20 00 15 84 00 01   ........ .. .....
0060   00 12 0a 10 02 00 20 00   15 84 00 02 00 12 0a 10   ...... . ........
0070   02 00 20 00 15 84 00 03   00 12 0a 10 02 00 20 00   .. ..... ...... .
0080   15 84 00 04 00 12 0a 10   02 00 04 00 15 84 00 05   ........ ........
0090   00
[1]frame[-5:5]==15:84:00:05:00

0030   0b 68 39 95 00 00 03 00   00 5b 02 f0 80 32 01 00   .h9..... .[...2..
0040   00 bd 56 00 4a 00 00 04   06 12 0a 10 02 00 20 00   ..V.J... ...... .
0050   1a 84 00 00 00 12 0a 10   02 00 20 00 1a 84 00 01   ........ .. .....
0060   00 12 0a 10 02 00 20 00   1a 84 00 02 00 12 0a 10   ...... . ........
0070   02 00 20 00 1a 84 00 03   00 12 0a 10 02 00 20 00   .. ..... ...... .
0080   1a 84 00 04 00 12 0a 10   02 00 18 00 1a 84 00 05   ........ ........
0090   00                                                  .
[2]frame[-5:5]==1a:84:00:05:00

0030   0b 68 92 be 00 00 03 00   00 5b 02 f0 80 32 01 00   .h...... .[...2..
0040   00 93 ff 00 4a 00 00 04   06 12 0a 10 02 00 20 00   ....J... ...... .
0050   1a 84 00 00 c0 12 0a 10   02 00 20 00 1a 84 00 01   ........ .. .....
0060   c0 12 0a 10 02 00 20 00   1a 84 00 02 c0 12 0a 10   ...... . ........
0070   02 00 20 00 1a 84 00 03   c0 12 0a 10 02 00 20 00   .. ..... ...... .
0080   1a 84 00 04 c0 12 0a 10   02 00 18 00 1a 84 00 05   ........ ........
0090   c0
[3]frame[-5:5]==1a:84:00:05:c0
```

S7-400 Tag Logging, Alarm Logging deaktiv

```
0030   0b 68 7f dd 00 00 03 00   00 5d 02 f0 80 32 01 00    .h...... .]...2..
0040   00 f3 02 00 4c 00 00 04   01 12 48 b0 0e 20 00 15    ....L... ..H.. ..
0050   00 00 20 00 15 00 20 20   00 15 00 40 20 00 15 00    .. ...     ...@ ...
0060   60 20 00 15 00 80 20 00   15 00 a0 20 00 15 00 c0    ` .... . ... ....
0070   20 00 15 00 e0 20 00 15   01 00 20 00 15 01 20 20      .... .. .. ...
0080   00 15 01 40 20 00 15 01   60 10 00 15 01 80 20 00    ...@ ... `..... .
0090   16 00 00                                                  ...
[4]frame[-5:5]==20:00:16:00:00
                                          ...
0030   0b 68 4a 3a 00 00 03 00   00 5d 02 f0 80 32 01 00    .hJ:.... .]...2..
0040   00 84 10 00 4c 00 00 04   01 12 48 b0 0e 20 00 1a    ....L... ..H.. ..
0050   00 00 20 00 1a 00 20 20   00 1a 00 40 20 00 1a 00    .. ...     ...@ ...
0060   60 20 00 1a 00 80 20 00   1a 00 a0 20 00 1a 00 c0    ` .... . ... ....
0070   20 00 1a 00 e0 20 00 1a   01 00 20 00 1a 01 20 20      .... .. .. ...
0080   00 1a 01 40 20 00 1a 01   60 10 00 1a 01 80 20 00    ...@ ... `..... .
0090   1b 00 00                                                  ...
[5]frame[-5:5]==20:00:1b:00:00

0030   0b 68 1d ad 00 00 03 00   00 5d 02 f0 80 32 01 00    .h...... .]...2..
0040   00 80 cf 00 4c 00 00 04   01 12 48 b0 0e 20 00 1a    ....L... ..H.. ..
0050   00 a0 20 00 1a 00 c0 20   00 1a 00 e0 20 00 1a 01    .. ....    .... ...
0060   00 20 00 1a 01 20 20 00   1a 01 40 20 00 1a 01 60    . ...    . ..@ ...`
0070   10 00 1a 01 80 20 00 1b   00 00 20 00 1b 00 20 20    ..... .. .. ...
0080   00 1b 00 40 20 00 1b 00   60 20 00 1b 00 80 20 00    ...@ ... ` .... .
0090   1b 00 a0                                                  ...
[6]frame[-5:5]==20:00:1b:00:a0
```

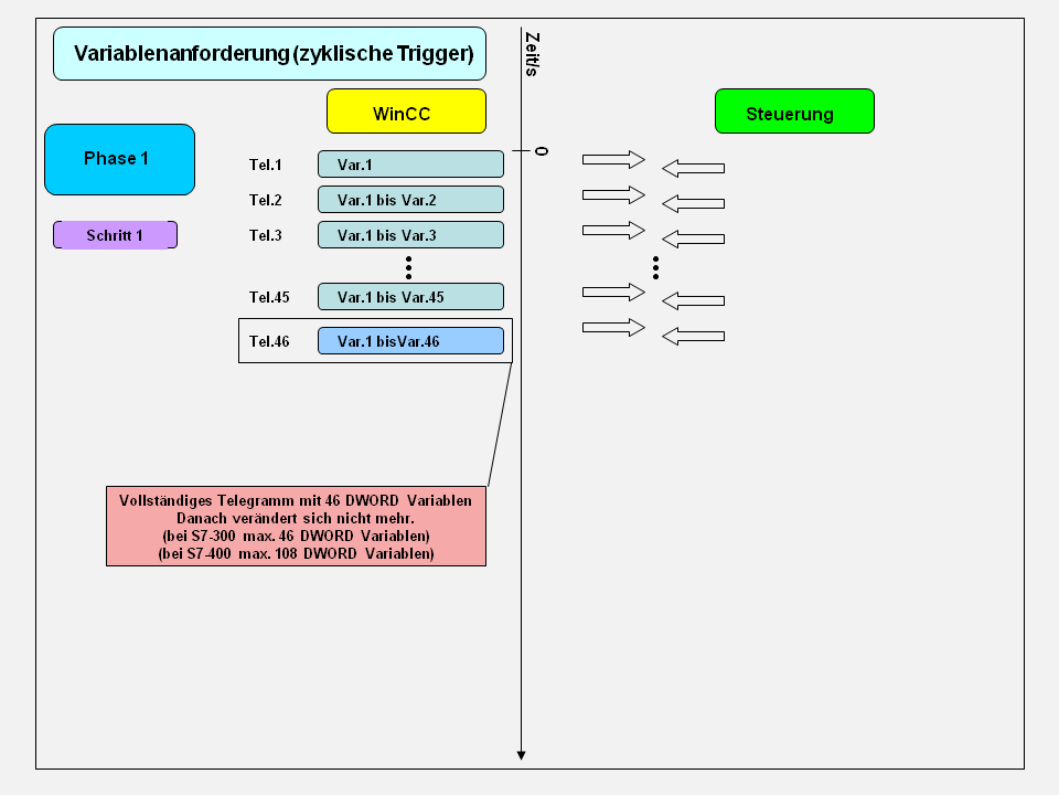
S7-300 Tag Logging, Alarm Logging aktiv

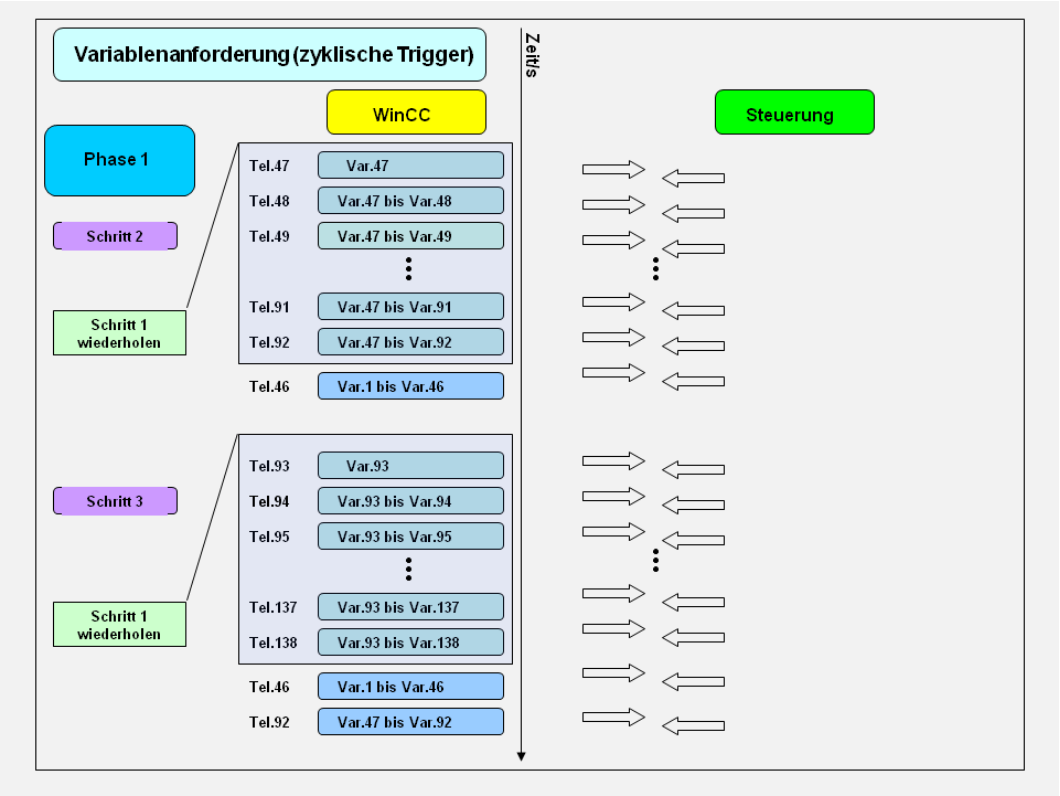
```
0030   0b 68 d1 2f 00 00 03 00   00 5b 02 f0 80 32 01 00    .h./.... .[...2..
0040   00 0e 19 00 4a 00 00 04   06 12 0a 10 02 00 20 00    ....J... ...... .
0050   15 84 00 00 00 12 0a 10   02 00 20 00 15 84 00 01    ........ .. .....
0060   00 12 0a 10 02 00 20 00   15 84 00 02 00 12 0a 10    ...... . ........
0070   02 00 20 00 15 84 00 03   00 12 0a 10 02 00 20 00    .. ..... ...... .
0080   15 84 00 04 00 12 0a 10   02 00 18 00 15 84 00 05    ........ ........
0090   00
[7]frame[-5:5]==15:84:00:05:00

0030   0b 68 ee 25 00 00 03 00   00 5b 02 f0 80 32 01 00    .h.%.... .[...2..
0040   00 ea 26 00 4a 00 00 04   06 12 0a 10 02 00 20 00    ..&.J... ...... .
0050   15 84 00 00 20 12 0a 10   02 00 20 00 15 84 00 01    .... ... .. .....
0060   20 12 0a 10 02 00 20 00   15 84 00 02 20 12 0a 10      ..... . .... ...
0070   02 00 20 00 15 84 00 03   20 12 0a 10 02 00 20 00    .. .....   ..... .
0080   15 84 00 04 20 12 0a 10   02 00 18 00 15 84 00 05    .... ... ........
0090   20
[8]frame[-5:5]==15:84:00:05:20

0030   0b 68 e0 2f 00 00 03 00   00 5b 02 f0 80 32 01 00    .h./.... .[...2..
0040   00 38 2d 00 4a 00 00 04   06 12 0a 10 02 00 20 00    .8-.J... ...... .
0050   1a 84 00 00 20 12 0a 10   02 00 20 00 1a 84 00 01    .... ... .. .....
0060   20 12 0a 10 02 00 20 00   1a 84 00 02 20 12 0a 10      ..... . .... ...
0070   02 00 20 00 1a 84 00 03   20 12 0a 10 02 00 20 00    .. .....   ..... .
0080   1a 84 00 04 20 12 0a 10   02 00 18 00 1a 84 00 05    .... ... ........
0090   20
[9]frame[-5:5]==1a:84:00:05:20

0030   0b 68 56 e6 00 00 03 00   00 5b 02 f0 80 32 01 00    .hV..... .[...2..
0040   00 e4 40 00 4a 00 00 04   06 12 0a 10 02 00 20 00    ..@.J... ...... .
0050   1a 84 00 00 e0 12 0a 10   02 00 20 00 1a 84 00 01    ........ .. .....
0060   e0 12 0a 10 02 00 20 00   1a 84 00 02 e0 12 0a 10    ...... . ........
0070   02 00 20 00 1a 84 00 03   e0 12 0a 10 02 00 20 00    .. ..... ...... .
0080   1a 84 00 04 e0 12 0a 10   02 00 18 00 1a 84 00 05    ........ ........
0090   e0
[10]frame[-5:5]==1a:84:00:05:e0
```

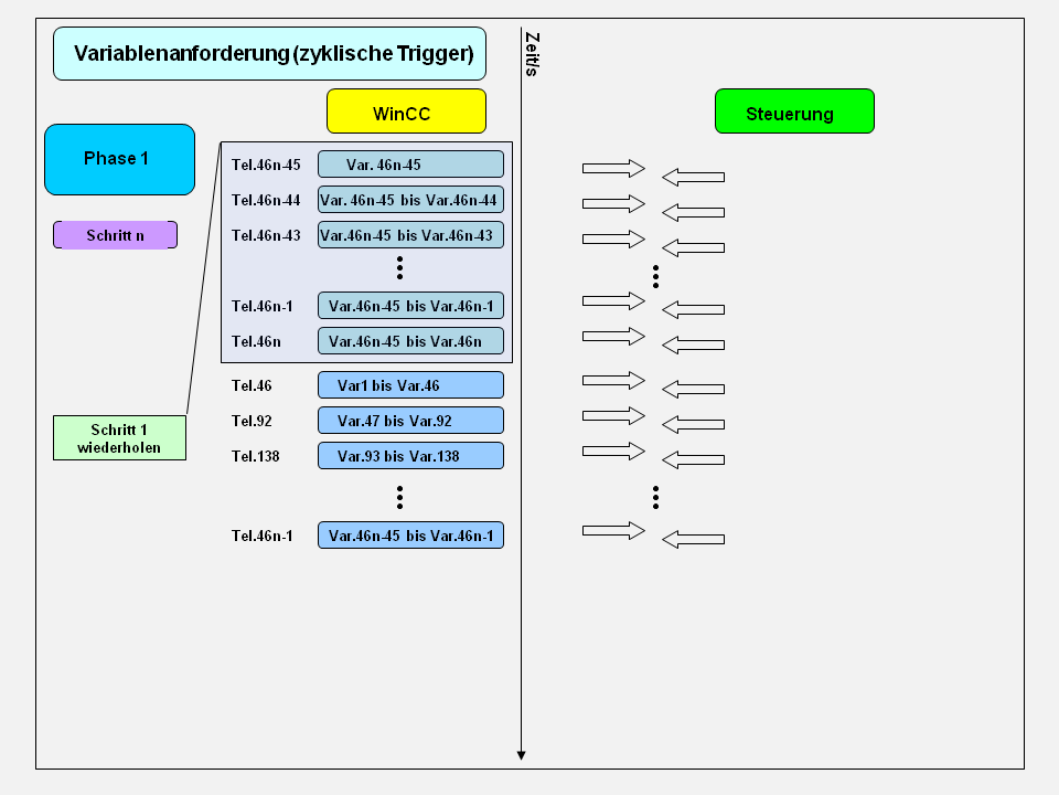
S7-400 Tag Logging, Alarm Logging aktiv

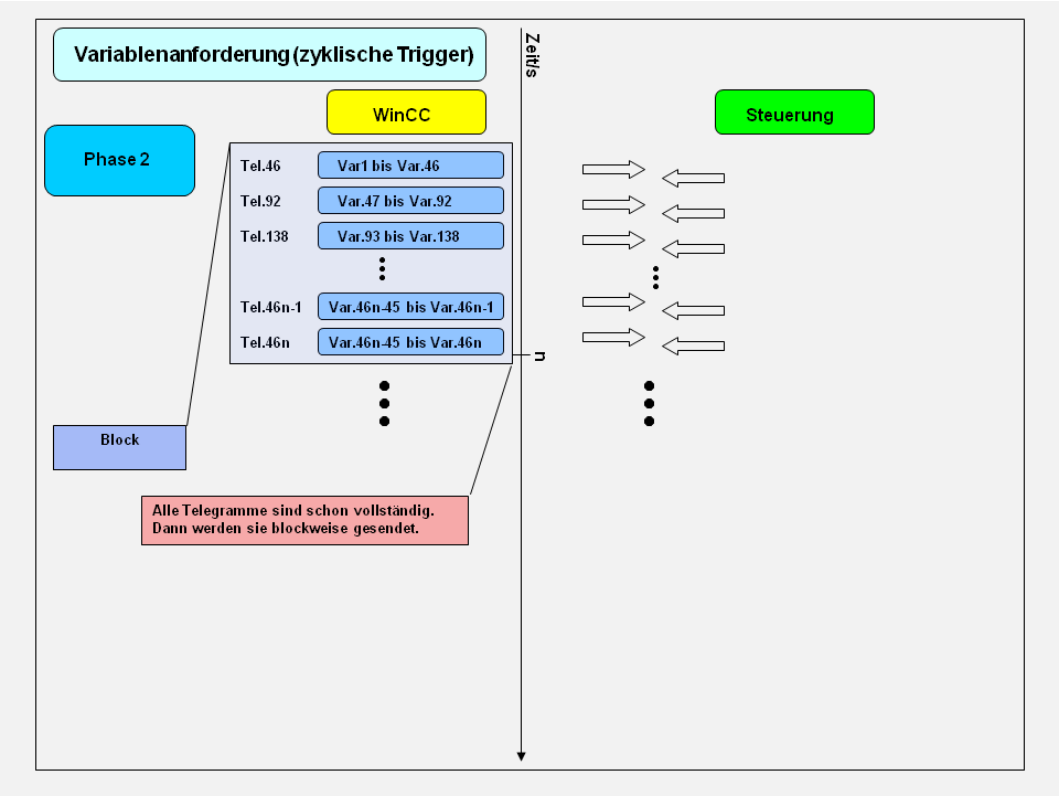
```
0030   0b 68 26 74 00 00 03 00   00 5d 02 f0 80 32 01 00    .h&t.... .]...2..
0040   00 22 20 00 4c 00 00 04   01 12 48 b0 0e 20 00 16    ." .L... ..H.. ..
0050   00 1c 20 00 16 00 3c 20   00 16 00 5c 20 00 16 00    .. ...<   ...¥ ...
0060   7c 20 00 16 00 9c 20 00   16 00 bc 20 00 16 00 dc    | .... . ... ....
0070   20 00 16 00 fc 20 00 16   01 1c 20 00 16 01 3c 20     .... .. .. ...<
0080   00 16 01 5c 14 00 16 01   7c 20 00 17 00 00 1c 00    ...¥.... | ......
0090   17 00 20
[11]frame[-5:5]==1c:00:17:00:20

0030   0b 68 b6 a8 00 00 03 00   00 5d 02 f0 80 32 01 00    .h...... .]...2..
0040   00 8b 2e 00 4c 00 00 04   01 12 48 b0 0e 20 00 16    ....L... ..H.. ..
0050   00 20 20 00 16 00 40 20   00 16 00 60 20 00 16 00    .   ...@   ...` ...
0060   80 20 00 16 00 a0 20 00   16 00 c0 20 00 16 00 e0    . .... . ... ....
0070   20 00 16 01 00 20 00 16   01 20 20 00 16 01 40 20     .... .. .   ...@
0080   00 16 01 60 10 00 16 01   80 20 00 17 00 00 20 00    ...`.... . .... .
0090   17 00 20
[12]frame[-5:5]==20:00:17:00:20

0030   0b 68 0f 76 00 00 03 00   00 5d 02 f0 80 32 01 00    .h.v.... .]...2..
0040   00 7a de 00 4c 00 00 04   01 12 48 b0 0e 20 00 1b    .z..L... ..H.. ..
0050   00 1c 20 00 1b 00 3c 20   00 1b 00 5c 20 00 1b 00    .. ...<   ...¥ ...
0060   7c 20 00 1b 00 9c 20 00   1b 00 bc 20 00 1b 00 dc    | .... . ... ....
0070   20 00 1b 00 fc 20 00 1b   01 1c 20 00 1b 01 3c 20     .... .. .. ...<
0080   00 1b 01 5c 14 00 1b 01   7c 20 00 1c 00 00 1c 00    ...¥.... | ......
0090   1c 00 20
[13]frame[-5:5]==1c:00:1c:00:20

0030   0b 68 e1 a3 00 00 03 00   00 5d 02 f0 80 32 01 00    .h...... .]...2..
0040   00 f7 e1 00 4c 00 00 04   01 12 48 b0 0e 20 00 1b    ....L... ..H.. ..
0050   00 bc 20 00 1b 00 dc 20   00 1b 00 fc 20 00 1b 01    .. ....   .... ...
0060   1c 20 00 1b 01 3c 20 00   1b 01 5c 14 00 1b 01 7c    . ...< . ..¥....|
0070   20 00 1c 00 00 20 00 1c   00 20 20 00 1c 00 40 20     .... .. .   ...@
0080   00 1c 00 60 20 00 1c 00   80 20 00 1c 00 a0 1c 00    ...` ... . ......
0090   1c 00 c0
[14]frame[-5:5]==1c:00:1c:00:c0
```

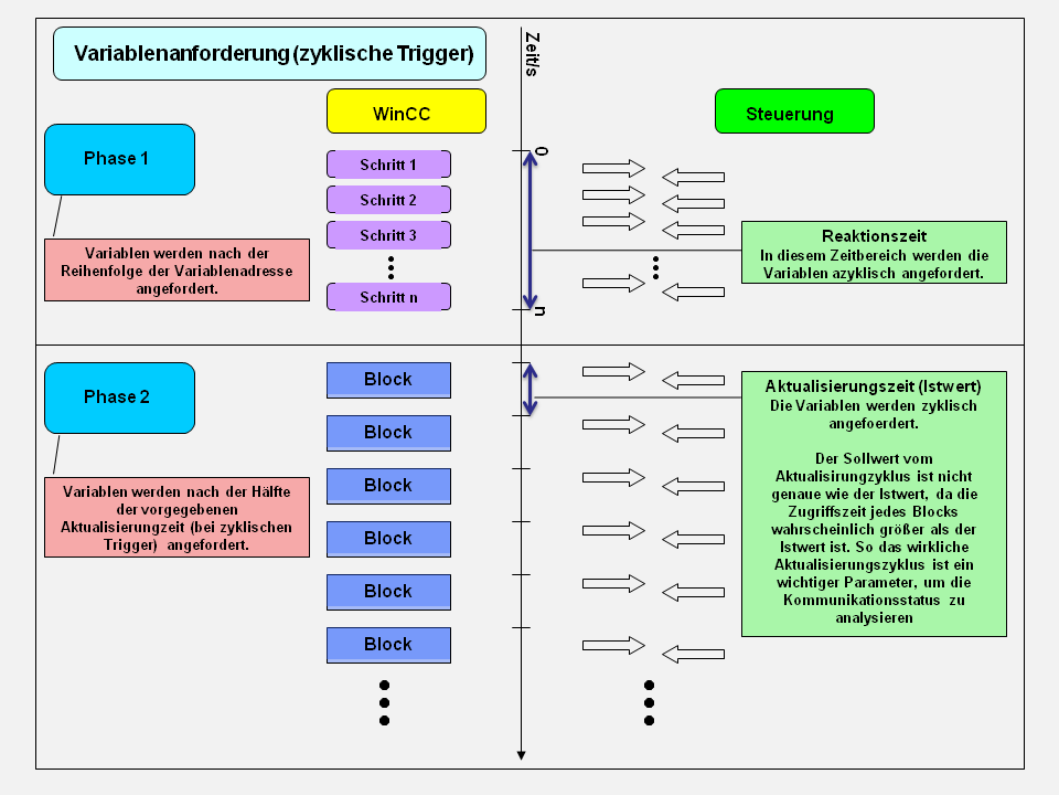
A [05] Ablauf der Variablenanforderung bei zyklischen Trigger und Variablentrigger

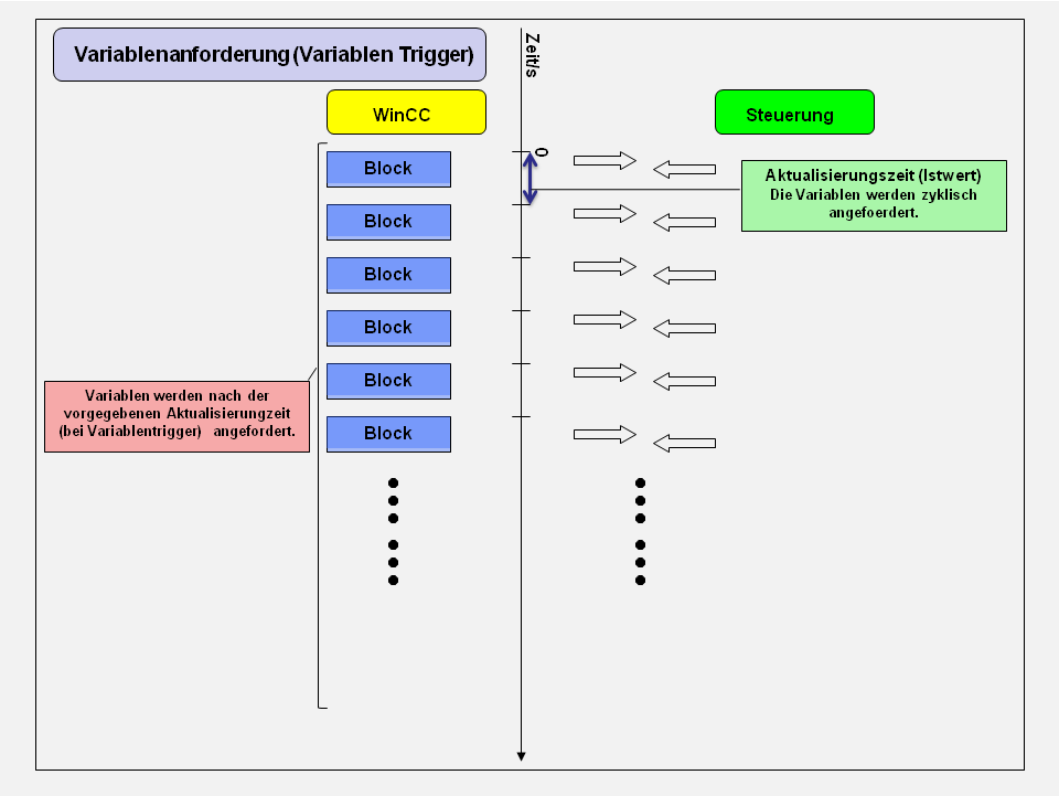





Variablenanforderung (zyklische Trigger)
Zeit/s
WinCC
Steuerung
Phase 1
Schritt n
Tel.46n-45
Var. 46n-45
Tel.46n-44
Var. 46n-45 bis Var.46n-44
Tel.46n-43
Var.46n-45 bis Var.46n-43
Tel.46n-1
Var.46n-45 bis Var.46n-1
Tel.46n
Var.46n-45 bis Var.46n
Tel.46
Var1 bis Var.46
Tel.92
Var.47 bis Var.92
Tel.138
Var.93 bis Var.138
Tel.46n-1
Var.46n-45 bis Var.46n-1
Schritt 1 wiederholen


Variablenanforderung (zyklische Trigger)
Zeit/s
WinCC
Steuerung
Phase 2
Tel.46
Var1 bis Var.46
Tel.92
Var.47 bis Var.92
Tel.138
Var.93 bis Var.138
Tel.46n-1
Var.46n-45 bis Var.46n-1
Tel.46n
Var.46n-45 bis Var.46n
n
Block
Alle Telegramme sind schon vollständig.
Dann werden sie blockweise gesendet.


Variablenanforderung (zyklische Trigger)
Zeit/s
WinCC
Steuerung
Phase 1
Schritt 1
Schritt 2
Schritt 3
Schritt n
0
n
Variablen werden nach der Reihenfolge der Variablenadresse angefordert.
Reaktionszeit
In diesem Zeitbereich werden die Variablen azyklisch angefordert.
Phase 2
Block
Block
Block
Block
Block
Block
Variablen werden nach der Hälfte der vorgegebenen Aktualisierungzeit (bei zyklischen Trigger) angefordert.
Aktualisierungszeit (Istwert)
Die Variablen werden zyklisch angefoerdert.
Der Sollwert vom Aktualisirungzyklus ist nicht genaue wie der Istwert, da die Zugriffszeit jedes Blocks wahrscheinlich größer als der Istwert ist. So das wirkliche Aktualisierungszyklus ist ein wichtiger Parameter, um die Kommunikationsstatus zu analysieren


Variablenanforderung (Variablen Trigger)
Zeit/s
WinCC
Steuerung
Block
Block
Block
Block
Block
Block
0
Aktualisierungszeit (Istwert)
Die Variablen werden zyklisch angefoerdert.
Variablen werden nach der vorgegebenen Aktualisierungzeit (bei Variablentrigger) angefordert.

Eigenständigkeitserklärung

Hiermit erkläre ich, dass ich die vorliegende Arbeit selbstständig und nur unter Verwendung der angegebenen Literatur und Hilfsmittel angefertigt habe. Stellen, die wörtlich oder sinngemäß aus Quellen entnommen wurden, sind als solche kenntlich gemacht. Diese Arbeit wurde in gleicher oder ähnlicher Form noch keiner anderen Prüfungsbehörde vorgelegt.

Nürnberg, 27.02.2013 Min Xie